ARI Research Note 96-06

Theories, Methods, and Tools for the Design of User-Centered Computer Systems

Gerhard Fischer, Walter Kintsch, Evelyn Fersti, Peter Foltz,
Scott Henninger, David Redmiles, and Curt Stevens
University of Colorado

Research and Advanced Concepts Office
Michael Drillings, Acting Chief

October 1995

United States Army
Research Institute for the Behavioral and Social Sciences

U.S. ARMY RESEARCH INSTITUTE
FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency Under the Jurisdiction of the Deputy Chief of Staff for Personnel

EDGAR M. JOHNSON
Director

Research accomplished under contract
for the Department of the Army

University of Colorado

Technical review by

Joseph Psotka

NOTICES

DISTRIBUTION: This report has been cleared for release to the Defense Technical Information Center (DTIC) to comply with regulatory requirements. It has been given no primary distribution other than to DTIC and will be available only through DTIC or the National Technical Information Service (NTIS).

FINAL DISPOSITION: This report may be destroyed when it is no longer needed. Please do not return it to the U.S. Army Research Institute for the Behavioral and Social Sciences.

NOTE: The views, opinions, and findings in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other authorized documents.

REPORT DOCUMENTATION PAGE

1. REPORT DATE 1995, October	2. REPORT TYPE Final	3. DATES COVERED July 1986
4. TITLE AND SUBTITLE Theories, Methods, and Tools for the Design of User-Centered Computer Systems		5a. CONTRACT OR GRANT NUMBER MDA903-86-C-0143
		5b. PROGRAM ELEMENT NUMBER 0601102A
6. AUTHOR(S) Gerhard Fischer, Walter Kintsch, Evelyn Fersti, Peter Foltz, Scott Henninger, David Redmiles, and Curt Stevens (University of Colorado)		5c. PROJECT NUMBER B74F
		5d. TASK NUMBER 1901
		5e. WORK UNIT NUMBER C10
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Department of Computer Science and Psychology and Institute of Cognitive Science University of Colorado Boulder, CO 80309		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) U.S. Army Research Institute for the Behavioral and Social Sciences ATTN: PERI-BR 5001 Eisenhower Avenue Alexandria, VA 22333-5600		10. MONITOR ACRONYM ARI
		11. MONITOR REPORT NUMBER Research Note 96-06
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		
13. SUPPLEMENTARY NOTES COR: George Lawton		

14. ABSTRACT *(Maximum 200 words)*:

The goal of this research at the general level is to develop theories, methods, and tools for the design of user-centered computer systems, and at the specific level to design, implement, and evaluate a customizable Personalized Intelligent Retrieval system. Our research is based on the basic hypothesis that the following duality exists: (1) user-centered system design cannot be done and understood without trying to test existing ones, extending existing ones, and designing new ones, and (2) user-centered system design cannot be understood by just doing it. The system building efforts must be based on a deep understanding of the theoretical and methodological issues behind them, derived primarily from cognitive science, and, as far as evaluation is concerned, from human factors/cognitive ergonomics.

15. SUBJECT TERMS

User-centered computer systems	Situation models	System models
Information retrieval	Computing tasks	Learning

SECURITY CLASSIFICATION OF			19. LIMITATION OF ABSTRACT	20. NUMBER OF PAGES	21. RESPONSIBLE PERSON (Name and Telephone Number)
16. REPORT Unclassified	17. ABSTRACT Unclassified	18. THIS PAGE Unclassified	Unclassified	58	

Table of Contents

List of Figures

1. Summary

1.1 Objectives

The objectives of the project were stated in the original proposal as follows:

> The goal of this research at the general level is to develop theories, methods, and tools for the design of user-centered computer systems, and at the specific level to design, implement, and evaluate a customizable Personalized Intelligent Retrieval System. Our research is based on the basic hypothesis that the following duality exists: (1) user-centered system design cannot be done and understood without trying to test existing ones, extend existing ones, and design new ones, and (2) user-centered system design cannot be understood by just doing it; the system building efforts must be based on a deep understanding of the theoretical and methodological issues behind them, derived primarily from cognitive science, and, as far as evaluation is concerned, from human factors / cognitive ergonomics.

This duality required an evolutionary approach towards system design and evaluation. During this evolution, our integration of research at the conceptual level inevitably led to an integration of research at the system building level. In this manner we approached our goal: to design, implement, and evaluate customizable, *personalized information environments* [Fischer, Nieper 87]. These systems instantiated our progress in achieving our goals, and raised numerous theoretical and psychological issues providing new research topics to be investigated in future work.

Figure 1-1 provides an overview of the research activities conducted in the project. We have concentrated on the general domain of information management. The problem of information management is not the availability of information, but the ability of humans to process it. Information overload occurs when the amount of available information is so large that the demands on our time required to find relevant information and process it overwhelm our abilities. When information needs require people to choose from vast repositories of information, they encounter additional problems. These problems range from the noise caused by the diversity of available information to the onset of boredom caused by the task of filtering out that noise. In addition, overload conditions effect strategies that information seekers choose to employ in the filtering task. As overload increases, the effectiveness of strategies also tends to decrease. Finally, the tendency for an information source to exhibit information overload can directly affect peoples' willingness to access that source.

In addition to investigating the sources of information overload it is important to look for solutions. The crux of any strategy for reducing the effects of information overload is lowering the amount of overload to levels at which effective manual filtering can take place. It is likely that successful systems for accessing large amounts of information will have to employ several strategies simultaneously.

The theories and systems discussed below investigate the issues of information management in a variety of domains, interchanging ideas among all. The domains investigated include planning routine computing tasks, learning from both normal text and hypertext documents, retrieving literature citations, and retrieving and understanding software objects.

1.2 Overview of this Report

Following this introductory summary, Chapter 2 gives an overview of the theoretical and conceptual work. Chapter 3 provides details of the use and formation of situation models. Chapter 4 describes the design, implementation, and evaluation of systems we built. Although we have chosen to represent the results of the project in separate chapters, the driving force behind many research activities in the project was the

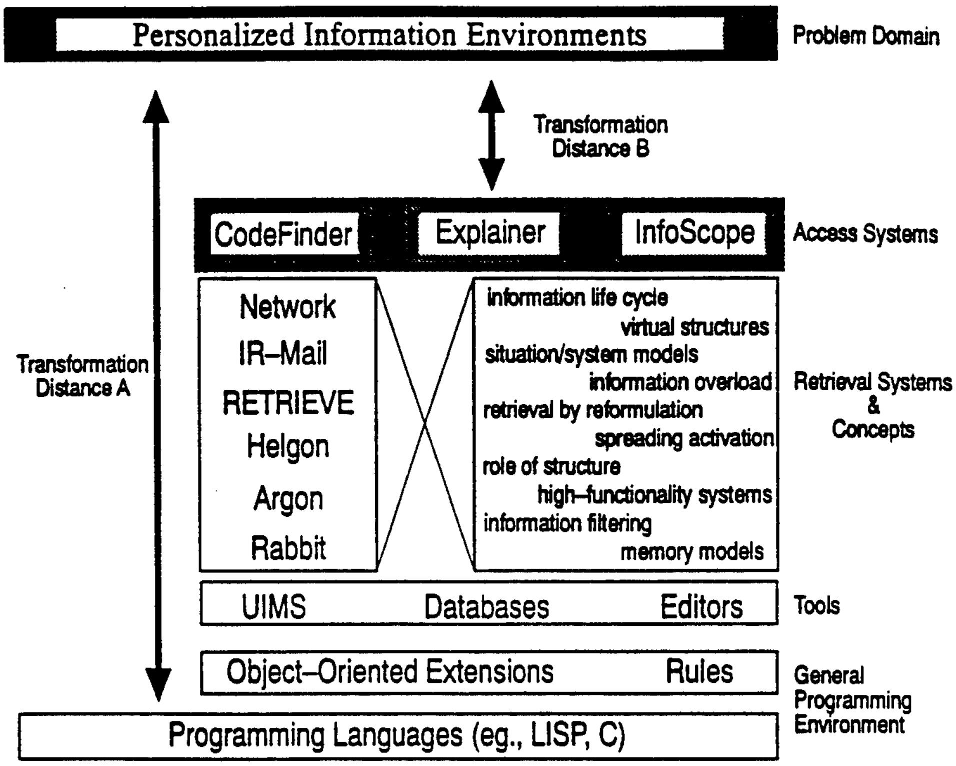


Figure 1-1: Project History: Theories and Systems

In order to reduce the transformation distances between standard programming languages and personalize information environments, we support these environments by a layered architecture. Each layer utilizes stable subsystems and demonstrated concepts from the layer below. Through this incremental approach, systems are realized that operate in problem domains, as opposed to computer system domains.

---

tight integration between cognitive science research and innovative system building. Chapter 5 discusses applications, collaborations, and research issues raised for future investigation.

Appendix I lists our publication record. Appendix II describes gives information about graduate students who do research for the project. Appendix III contains additional information about the project. Finally, Appendix IV documents two assessments of our research efforts by researchers who are simultaneously familiar with the needs of the Army as well as our work: (1) Thomas W. Mastaglio, a former Lieutenant Colonel in the U.S. Army, and (2) James Sullivan, currently a Major in the U.S. Army.

> **Remark:** In order to be brief, this report is focused on the work done since the Interim Report which was made available to ARI in March 1989.

2. Theoretical and Conceptual Research

2.1 User-Centered System Design

There are many aspects to the term "user-centered" [Norman, Draper 86; Norman 93]. We have used the concept of personalized information environments to focus on the individual's needs and role in interacting with computer systems for information management [Fischer, Nieper 87]. Our theoretical starting point was a distinction between two levels of mental representations users have of the tasks they want to perform with computer systems, the *situation model* and the *system model* [Turner 88]. The situation model is a representation of the task a computer user wants to perform and is in terms specific to the task domain. It is subjective and varies somewhat among individuals, but our assumption has been that it is well specified (i.e., the users know what they want to do). In order to accomplish anything, however, the user's situation model must be transformed into a system model, which is normative and system specific.

The situation-system model distinction has been the driving idea behind the theorizing and system building in this project. Our question has been, how, for a variety of tasks in which information management plays a central role, this transformation from situation model to system model is achieved, and what system support can be provided for it. Figure 2-1 illustrates some possible alternatives for support.

2.2 Situation and System Models

The distinction between situation and systems models is not an ad hoc distinction, but is based on theoretical developments in other areas of research. The term "situation model" was introduced by van Dijk and Kintsch in 1983. Later, Kintsch and Greeno [1985] in their work on word arithmetic problems introduced the distinction between a person's understanding of the situation described in a word problem in everyday terms, and the mathematization of that situation (there called the "arithmetic problem model"). The term "system model" in the present work corresponds to the problem model in the mathematics word problem domain. The situation and system models are related to what a large number of researchers in cognitive science refer to as "mental models:" both situation and system model are a type of mental model. The Breckenridge Workshop in 1988 [Turner 88] was devoted to a systematic exploration of the relevant issues.

Below, two aspects of our project will be described that have to do with the question how situation models are to be transformed into workable system models. Both concern information management. The first aspect is a theoretical approach to examining the role of the situation model. This work studies how to construct an appropriate system model, given a situational understanding of routine computing tasks. This work has taken primarily an experimental and theoretical focus, but possible future developments for the design of help systems incorporating some of our findings will be discussed. The second aspect of the project describes systems that make the system model transparent — the HELGON system and its descendants. It involves a large system building effort, experimental evaluations, and theory-based modifications and developments. This effort has incorporated knowledge gained from the theoretical approach in order to help bridge the gulf between situation and system models.

Situational understanding of routine cognitive tasks. An understanding of a user's situation model of a system permits an understanding of what information a user currently has, and how new information will be incorporated into the situation model. By modeling a situation model, we can make some explicit predictions about a user's success on a particular system. We have described in earlier work a theoreti-

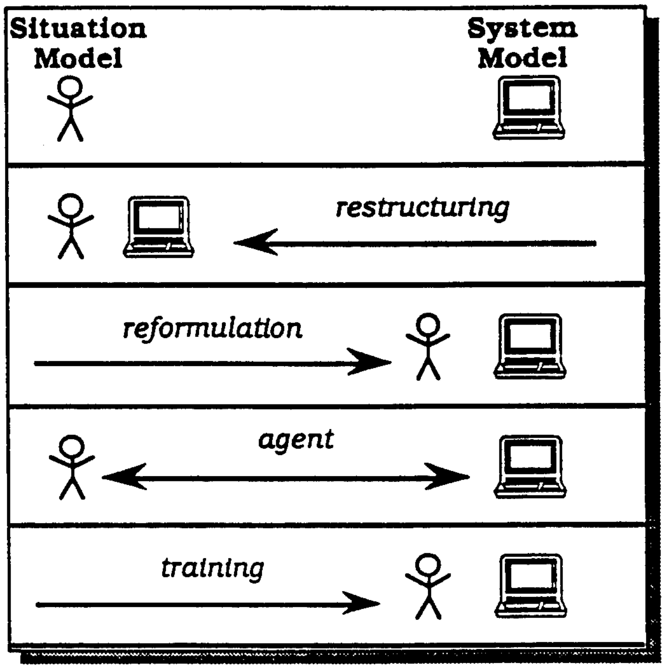


Figure 2-1: Bridging the Gap between Situation and System Model

The situation model-system model distinction supports the analysis of the gap between what users *do* know and what what they *must* know in order to use and understand systems. There are several methods for bridging this gap, each filling a distinct their own role in supporting system use.

- The first row illustrates the situation where there is no support for bridging the gap.
- The second row shows the approach in which a new system model is constructed that is closer to an individual's situation model and hence easier to understand. This approach is pursued in the INFOSCOPE and CODEFINDER systems.
- The third row illustrates the possibility of making the system model more transparent; allowing users to express their situation model incrementally within the system model. This approach is pursued in the HELGON and CODEFINDER systems.
- The fourth row shows how an agent can help translate a query from the situation into the system model. This approach is pursued in the INFOSCOPE system.
- The last row illustrates the training approach, where users are trained to express themselves directly in the system model. This is not appropriate for situations where the tasks/interests/queries change since these changes cannot be trained. This is why restructuring and reformulation must augment training. These technologies recognize the existence of a specific problem context.

---

cal account of how experts form problem models in a familiar domain — plans for routine computing tasks [Mannes, Kintsch 91].

Additional work has been performed in an empirical study of how situation models are modified through learning additional information [Ferstl 91]. This research has studied the changes that take place in knowledge structures when new information is added into a situation model. A simulation permits predictions of the knowledge structures obtained after reading information. The development of an appropriate situation model depends crucially on the previous knowledge of the comprehender or problem solver, as well as the text input or the problem statement. Successful integration of these two components permits

the learning of new facts and the reorganizing of knowledge structures according to new information. Results from this study support the hypothesis that the episodic text memory and the previous knowledge structures were integrated in the situational representation. This research provides a model of how new information is integrated into the situation model and further understanding of the role of prior knowledge in developing a situation model.

Navigating an information space: the role of a situational representation. The situational representation can play a major role in navigating large information spaces. In navigating large information spaces, the user must have a good model of the role of the relationships between the disparate pieces of information. Thus, prior knowledge of the information space can affect the user's ability to find the desired information. If a user has no prior knowledge then the user may be unable to use the structures of the information space appropriately. This typically results in information overload or feelings of being "lost in hyperspace."

One approach to aid users with little understanding of the information space is to develop automatic methods of providing background context. Foltz [Foltz 91a] has applied the VanDijk & Kintsch [Dijk, Kintsch 83] model in order to determine places in an information space in which users with little background knowledge will encounter difficulties in using the information space. The model can then predict what information the user may be missing and automatically provide it to the user. This therefore can provide support for users of a system when they have an impoverished situational representations.

2.3 Query Construction and Relevance Evaluation

Traditional information retrieval research assumes that a well-articulated query can be easily thought out and concentrates primarily on retrieval efficiency [Belkin, Croft 87]. Although this assumption works in well-understood domains, it does not apply to ill-defined problem domains, in which users need to elaborate and experiment with a problem before the information need is fully identified. Defining the problem is a large part of the problem, and support is needed for an incremental process of exploring the information space while refining the query.

For instance, in the domain of literature citations, the relevance of retrieved information can be judged easily by users. However, in a more complex domain, such as software objects (e.g., functions, subroutines, or programs), comprehending the retrieved objects becomes a significant problem. The literature citations use a familiar form of language that allows the gap between the situation and system model to be rather small. This is not the case with software objects, in which users may have problems understanding the language, abstractions, and interdependencies upon which the software object is built. This causes the gap between situation and system models to be large enough to require support for judging whether the item meets the information need.

Intertwining location and comprehension. In a cooperative problem solving environment, judging the relevance of an object is supported by intertwining the processes of location and comprehension. Once an object is retrieved, users may study it to determine its appropriateness for their task. Once an understanding is achieved, users are in a better position to understand their needs and can make refined attempts at location. The integration of location and comprehension in the process of retrieval and use of information is discussed in more detail in a later section about systems building, Chapter 4.

Retrieval by reformulation versus relevance feedback. Another way of intertwining location and com-

prehension is through a query formulation technique known as retrieval by reformulation [Williams 84; Williams et al. 82]. Users incrementally develop a formal query by critiqing examples resulting from intermediate and partially formed queries [Fischer, Nieper-Lemke 89; Fischer, Henninger, Redmiles 91a].

Retrieval by reformulation differs from relevance feedback methods that attempt to automate the query reformulation process [Salton, Buckley 90]. Many relevance feedback systems work by having the user rate the relevance of items retrieved by an initial query. The ratings are then used to reformulate the query, often by adjusting the internal representation of the query and/or document representations (for example by adjusting link weights) to enhance retrieval around documents judged relevant and inhibit irrelevant areas [Belew 87; Oddy 77; Stanfill, Kahle 86]. Retrieval can follow immediately without any need for user intervention. However, analyses have shown that improvements in automatic methods bring about only small improvements in retrieval efficiency, whereas improvements in support for users' retrieval strategies, such as retrieval by reformulation, can bring about large improvements in retrieval efficiency [Foltz 91b; Salton, Buckley 90]. Retrieval by reformulation can be more efficient than relevance feedback because the mapping between a query and its result is explicit, facilitating the construction of a mental model of system behavior by users. Section 4 describes systems that develop further the ideas of retrieval by reformulation as well as incorporate relevance feedback.

Supporting users' situation and system models. In cooperative problem solving environments, the processes of location and comprehension of retrieved objects are closely coupled, supporting an interplay between query construction and relevance evaluation. The interplay helps users avoid a common pitfall in information management, not being able to formulate queries due either to poorly developed understanding of their problem domain or of the types of information they can access. As illustrated in Figure 2-1, users are helped to reformulate both their understanding of the problem (system model) and their understanding of a formal expression of a query, or problem, submitted to an information system (system model).

3. The Use and Formation of Situation Models

In the empirical and theoretical work which was part of the present project, we have investigated four aspects of the formation and use of situation models. First, Section 3.1 reviews and contrasts the use of situation models in human memory and information systems retrieval, providing a conceptual framework for the INFOSCOPE and IRMAIL systems discussed in Section 4 (see also [Foltz 92]) and for the HELGON system described in previous interim reports (see also [Fischer, Nieper 87; Fischer, Nieper-Lemke 89]). Second, Section 3.2 describes a computer simulation of peoples' situational understanding for planning routine action sequences in the domain of computing. Third, Section 3.3 describes a methodology and empirical study of how people form situation models when reading a text. Fourth, Section 3.4 reports on factors affecting the situational understanding of a hypertext contrasted to an ordinary linear text. A concluding subsection, Section 3.5, summarizes the current work and potential future research.

3.1 Use of Situation Models in Information Retrieval

A primary issue in information retrieval for both computer systems and humans is how the correct information can be accessed in an efficient manner. A recent estimate puts the amount of information accumulated by a normal adult over a lifetime to be on the order of 10^9 bits [Landauer 86]. Modern computers are also able to store about this same amount of information. While this great capacity permits humans to store many experiences and facts and computers to store many items of information, the key to the the flexibility of both the human brain and the computer is their ability to retrieve the relevant information quickly and accurately. Without fast access to information, both humans and computers could not perform the almost instantaneous actions for which they are both known. Similarly, accessing the wrong information would lead humans and computers to either perform the wrong actions or spend additional time retrieving the correct information.

Stages of Retrieval. The retrieval of information from both human memory and computer systems occurs in three stages: generating the retrieval cues, using the cues to retrieve information, and verifying that the retrieved information is what is desired. These stages differ though in the type of cognitive activity used. To retrieve information, cues first need to be generated. This process is strategic, involving controlled processing. A person develops cues that describe the information desired based on the current context. The generated cues are then used for the retrieval. In the case of information retrieval, the cues are transmitted to the computer, while in human retrieval, the cues are used automatically. In both cases, the actual retrieval process is automatic, i.e. not under a person's strategic control. Once information has been returned from the retrieval process, it is again under the control of a strategic process which evaluates the information to determine if it is what was desired.

The retrieval of information may not be just a single cycle of these three stages, but may involve several iterations. Information retrieved in a previous iteration may be added as additional cues for the next retrieval. While all three stages are important to retrieval, they differ greatly in the type of processes involved.

Human: automatic retrieval. In the memory literature, three classes of models have been developed. The compound cue model of memory [Raaijmaker, Shiffrin 81; Ratcliff, McKoon 88] assumes associative connections between retrieval cues and items in memory. The second type of models are the *spreading activation* models which use a semantic network of interconnected memory items (e.g., [Anderson 83]). Finally, distributed models of memory [Murdock 83; Hintzman 86] differ from the previous models in terms

of their representation of information in memory. These models describe rather well what happens when a given cue retrieves some associated information from memory although they make rather different assumptions about the underlying retrieval mechanism.

Human, strategic retrieval. While the models described above address retrieval issues once the cues have been provided, they do not address many of the issues of human strategic control of retrieval. When we need to retrieve information, we must decide what are the best cues to use for the retrieval and then evaluate what is returned to determine if it is what was sought. In a protocol study of people recalling the names of their high school classmates, Williams (1978) identified some of the strategies used in retrieval (see also [Walker, Kintsch 85]).

Retrieval is not just the process of searching a series of memory items given certain cues to see which items best match the cues. Instead, a context for the retrieval must be established. Individual cues, such as the few words used in memory models for cued recall, are seldom the only items used for retrieval. A person not only uses the word cue that was provided, but also much of the additional context of the situation in which the original word was encoded for the retrieval. As information is retrieved, it too serves as a context for retrieving additional information. Specific strategies are similarly used when initially encoding the information, thereby facilitating the use of context to help retrieve that same information.

Information systems: automatic retrieval. As in human memory retrieval, successful retrieval of an item depends on the similarity between that item and the cues provided. The variety of information retrieval models represent different methods of calculating and representing these similarities in order to maximize the effectiveness of the retrieval, given the tasks and environment. In the retrieval of textual information, each document is treated as a set of features, where each feature corresponds to a term used in the document. A standard method of retrieval is to create an inverted index in which each term is represented as a vector with each vector element representing whether a particular document contains the term. Given a query consisting of terms, the best matching documents can be retrieved through Boolean operations. Important examples of automatic retrieval mechanisms are the the vector space model [Salton, McGill 83] and the probabilistic retrieval model [Bookstein, Swanson 75].

The fact that many information retrieval methods require using the exact words used in the document to retrieve it highlights one of the deficiencies in current techniques. People seldom know which words will describe a document and there is a great variability in the choice of words between people. People choose the same single word to describe a familiar object only about 20% of the time [Furnas et al. 83]. Thus, keyword matching can fail due to polysemy (multiple meanings for a word) and synonymy (multiple ways of referring to one concept).

Information systems: strategic retrieval. In retrieval, it is often not clear to the user how or what can be retrieved. Users may not know which terms to use due to problems of synonymy and because they are not familiar with what type of information they can retrieve from the database. There are also problems with the actual interaction with the system; users may not know how to form a query or use the query language. Thus, a user interacting with a retrieval system may need to use some conscious strategies. To ease these strategic problems, information retrieval systems use methods for interpreting what a user wants and ways of letting the user browse through the information, such as relevance feedback [Salton, Buckley 90], information browsers, and hypertexts. Information browsers employ a set of rich connections between documents to allow a user to navigate through the space of information.

Integrated Models. Several models of information retrieval combine features from information retrieval systems and human memory. One of the primary ideas from the human memory literature that has been used in information retrieval is the concept of differential associative connections between items of information. For this reason, there have been a variety of retrieval models using some form of spreading activation. These models include the Memory Extender [Jones 86] and the connectionist retrieval system of Rose & Belew (1989). In these systems, a user can activate certain terms for their query. Activation then flows from these terms to the documents and other terms until the network settles. The most highly activated documents would then be retrieved.

Some information retrieval systems incorporate psychological models of users' retrieval strategies. One such system was William's [1984] RABBIT system based on his earlier research on strategies of memory retrieval. With RABBIT, people used the retrieval by reformulation technique, an iterative process of giving a partial description of what they wanted, retrieved a general context, and then used that information to narrow down the cues to get the information. In this manner, the system allowed users to do computer retrieval using a familiar memory retrieval strategy.

Personalized Information Environments. Humans are experts at using strategies to store and retrieve information from their own memory. They are familiar with the structure of the information stored and the retrieval cues that can be used since they did the initial encoding. This is not the case in computer retrieval. Users are seldom familiar with what information is available and how it is organized. This unfamiliarity hinders their ability to develop good retrieval cues to give to the system. Users are also not familiar with the ways of specifying the cues. Since most retrieval systems are term based, the exact terms must be specified to get the desired information.

Information retrieval systems are currently adding features to aid strategic retrieval: iterative retrieval (such as retrieval by reformulation), feedback, relevance of the degree of and browsing. Nevertheless, most systems do not make the large number of associations between information items that humans can. An improvement in information retrieval models can be made through tailoring the systems to incorporate greater semantic relationships in encoding and to use greater contextual information for retrieval such as user profiles. Spreading activation is one method for achieving this goal.

In the course of this project, the HELGON retrieval system [Fischer, Nieper-Lemke 89] led to a further development of these ideas. An empirical evaluation of HELGON was reported by Foltz & Kintsch [1988]. The addition of spreading activation to HELGON occurred in a system called Retrieve, developed as part of the present project by Fischer, Foltz, Kintsch, Nieper-Lemke, & Stevens [1989]. An extension of this system that combines spreading activation and retrieval by reformulation paradigms is described in Section 4.2 of this report (CodeFinder, [Henninger 91]). Thus, our work on retrieval systems and theory has been focused on facilitating the strategic aspects of retrieval, while taking advantage of the automatic spreading activation that characterizes human memory retrieval.

Psychological models of memory and of retrieval strategies highlight the current abilities of the human retrieval system and can provide directions for information retrieval systems to augment a person's ability to find information. Conversely, human memory retrieval can learn from the insights into computer information retrieval.

3.2 Use of Situation Models in Planning Routine Computing Tasks

When people act in a familiar domain, they "understand" what they have to do, what action to perform next. This kind of understanding is analogous to that used by people when they read stories. Readers "understand" why actors in a story behave in a certain way, why they do what they do. With this intuition, it was hypothesized that a model of text comprehension could be adapted to a model for planning actions. A cognitive simulation through a system called NETWORK verified this hypothesis.

The model of text comprehension used in NETWORK is the construction-integration model of Kintsch [Kintsch 88; Kintsch 92]. In this model, human memory is conceived of as an associative network with nodes standing for concepts and propositions representing knowledge from both the situation and system models. When an instruction is read, a set of symbolic production rules construct an associative network of interrelated items, a subset of long-term memory, specific to that task. These rules are weak in that they construct connections between items without respect for the current context or task at hand. This network is the basis for a second phase in which the integration takes place via connectionist constraint-satisfaction search. This process propagates activation throughout the network, serving to strengthen the connections between items which are consistent with each other and the context, and deactivating those items which initially were connected to others in the network, but are inconsistent.

NETWORK was used to simulate the planning of scriptal behavior for routine computing tasks. We first collected verbal protocols of experienced users acting out the instructional texts we wanted to study. After evaluating the verbal protocols three types of information could be identified: i) information subjects produced about their plans of action (or scripts) for the particular task; ii) meta-information where general knowledge (e.g. about computing and computers) played a role in the solution attempt; and iii) keystroke information, though this had no impact on our further investigation.

Both the plan of action and the meta-information were propositionalized according to standard procedures [Bovair, Kieras 85; Turner, Greene 78] and each proposition then became a node in the network representation of the domain. In this format each proposition is an atomic unit which contains a predicate and some number of arguments. For example, the propositionalization of the sentence 'Mike writes manuscripts', would appear as (WRITE MIKE MANUSCRIPT). (Note that propositions may also take as arguments other propositions, resulting in propositional embedding).

Plan information is described as a set of plan-element propositions, simple actions out of which entire plans can be synthesized. These are represented in an extended propositional format with three propositional fields: a name, preconditions, and outcomes. For example, the plan element to print a file is as
follows:

name:	(PRINTFILE)
preconditions:	(KNOW FILE LOCATION)
outcomes:	(EXIST HARDCOPY FILE)

Several rules were used to establish connections among items in the network. That is to assign the various weights relating propositions. To derive links between the meta-information propositions, certain linguistic relations, such as argument overlap and propositional embedding, were used. For example, the propositions (USE STUDENT MAIL) and (WRITE STUDENT PAPER) have a positive symmetric link between them because they share a reference to the concept STUDENT. These provide a crude approximation to the types of metrics people are hypothesized to use when comprehending a text [Kintsch,

[Dijk 78] and thus, when propositions share an argument or are embedded within one another a weight, which is one of several parameters to be set, specifying this relationship is entered in the matrix representing the network.

In addition, several relationships of a non-textual nature play a role in specifying the weights in the development of the network. In particular, items which are associated to each other, as determined by a **free** association study are related to each other with a particular weight. In this study, students were shown various phrases, such as "check your work," and were asked to write down the first thing that came to their minds. Similarities among subjects were identified and their responses were used as 'associated information' for NETWORK. Hence, an item such as (ISA UNIX SYSTEM) has a link to the item (USE PEOPLE COMPUTERS).

The plan elements that NETWORK uses to produce dynamically a plan of action are linked to the metapropositions via the aforementioned metrics and to one other in a more complex manner to form a causal chain. The causal chain is derived by assessing matches between the precondition and outcome fields of the plan elements' representations. Plan elements which provide as their outcome a precondition for another plan element, receive a link form that plan element. For example, a positive link exists from (DELETE FILE) to (FIND FILE). Conversely, a plan element which destroys a precondition for another plan element, receives an inhibitors link from that plan element. Hence there must be an inhibitors link from (FIND FILE) to (DELETE FILE). This arrangement allows for plan elements which provide essential preconditions to receive activation from the plan element requiring that state of affairs during the integration phase. Similarly, the inhibitors connections between plan elements allow for the flow of inhibition.

All this information is interrelated to form the system's long-term memory. This memory is used as a source of knowledge for all the tasks NETWORK can perform. In order to assess the functionality of the approach, simulations of several tasks were done.

A simulation in NETWORK involves the following steps. A propositional textbase is constructed for the instructional text to be understood and the current state of the world. Activation is then allowed to flow from this textbase into the long-term memory net, activating various part of that net differentially, depending on the content of the instruction. The specific plan element that becomes most strongly activated in this process is executed, assuming the preconditions are met (e.g., one can't delete a file if one does not know where that file is). The action changes the state of the world and allows activation to flow to the long-term memory net in a different pattern, activating some other plan element. Thus, there is a cycle of activation processes followed by actions which change the world, until the final action requested in the instructions "executed, resulting in that state of the world that was specified in the instructions.

Hence, NETWORK is not planning ahead and problem solving in the conventional sense. Instead, it understands the instruction in the context of the current situation and responds to it as well as it can, thereby - changing the situation. It then comprehends the new situation, and responds accordingly, repeating the cycle as necessary. NETWORK is therefore an example of situated cognition and provides a novel approach to planning. The possibilities and limits of that approach will be explored in future work.

3.3 Formation of Situation Models while Learning from Text

Text comprehension research has often focussed on how a text is understood and remembered. As part of the present project, we investigated how the information given in a text modifies the readers' knowledge structures. Part of the goal of the research was to evaluate the suitability of the applied methodology for evaluating the effects on long-term as well as episodic memory structures.

In Experiment I, 42 subjects read a story about a children's birthday party following one of either two instructions: to memorize it or to relate it to their own experiences. Before and immediately after hearing the story, a cued association task and a sorting task were administered to assess the associative organization between 60 key concepts of the text and its domain. The data were analyzed both qualitatively (using hierarchical clustering and network algorithms) and quantitatively. The reading instructions did not have a measurable impact on the results of the knowledge assessment tasks.

This study demonstrated the suitability of the knowledge assessment tasks for text comprehension research. Despite the relatively large number of items, most subjects perceived both sorting and cued association tasks as meaningful and straightforward. Since sorting yields symmetric association matrices and cued association asymmetric matrices (and thus a direct comparison of the two tasks was not performed), the results from the tasks gave supplementary information. Moreover, the data could be analyzed on different levels. Statistical comparisons of the structures before and after reading were possible for individual subjects, and qualitative analyses could be performed on amalgamated group data. The assessed knowledge structures before reading reflected the structure of the domain. For the sorting task, the words were organized according to a natural categorical structure, and for the cued association task, degree measures of the nodes in the associative structure corresponded to the centrality of the words in the birthday party script. After reading, text information was present in the structures. For both tasks, the number of text links (measured using proximity in the propositional structure of the text) increased significantly. Since this measure includes only connections directly mentioned in the text, the influence of text memory is even underestimated.

To summarize, cued association and sorting which mainly have been applied to measure the general world knowledge were shown also to be effective tools for assessing episodic text memory. The paradigm provide an informative way to assess the knowledge of subjects before and after reading. Knowledge was measured in terms of association strengths between previously selected words. The changes due to the intervening reading of a text could be directly measured both for individual subjects and groups of subjects. Not only does this paradigm allow analysis of text memory on different levels simultaneously, it also renders quantitative data which can be used in computational models. Since the tasks were easy to administer, time effective, and did not require a-priori assumptions about the expected changes in the structures, this method seems to be a promising tool for text comprehension research.

After having established the sensitivity of the knowledge assessment tasks for studying readers' situation models for a text Experiment II was conducted as a follow-up. The purpose of this study was threefold. First, it was aimed at replicating the previous results for the cued association task using more natural reading conditions. The sentence-by-sentence presentation on a computer screen was replaced by the presentation of the whole text typed on paper. Second, a control condition was added to provide a baseline for the reliability of the knowledge assessment tasks. Subjects in the control condition read an unrelated text instead of the birthday party story. Third, the cued association task was repeated after one week in order to measure the decay of the text memory influence after a delay.

As expected, the results of this experiment were similar to those of the previous study. The numbers of answers given which were from the list of selected words increased during the course of the experiment. The subjects in both conditions became more familiar with the list of words, and reading the related text had no impact on the size of this effect. The proportion of answers which corresponded to a link established in the text increased from 16% before reading to 40% after reading of the birthday party story, and decayed to 28% for the delayed test. For the control subjects, who read an unrelated text, this proportion stayed constant at about 1 7%. The large impact of the episodic text memory was also documented in the analysis of the overlap of subjects' networks at different test time. For the experimental group the highest overlap was found at the two test times after reading the birthday party story (After and Delay), and this overlap was mainly due to text links. In contrast, the highest overlap scores for the control group were found for the two test times in the first experimental session.

These tests demonstrated the suitability of the knowledge assessment paradigms are suitable for efficiently studying the interactions between general world knowledge and text information. A report on the two experiments described here is currently being prepared for publication [Ferstl, Kintsch 92].

3.4 Formation of Situation Models while Learning from Hypertext

Hypertext presents a way to read online text that is different than linear text. Although standard text is in linear form, hypertext is in the form of a semantic network of information in which a user may browse through parts of the text, jumping from one text node to another. This permits a reader to choose a path through the text that will be most relevant to his or her interests. Originally envisioned by Vanevar Bush in 1945 and first implemented by Engelbart in 1968, hypertext systems have now been developed for a variety of domains and tasks. Nevertheless, evaluations of hypertext systems have not been uniformly successful in showing that hypertext enhances human performance over linear text.

Researchers in the field of text comprehension have used user models to predict what will be learned from a text. These models have been successful at predicting such features as text comprehensibility, what features will be remembered from the text, and what inferences will be made from the text [Dijk, N'ntsch 83]. In this project we shall model the comprehension and goals of users of both a hypertext and a linear text using the Kintsch model of text comprehension [Kintsch 88; Kintsch 92; Dijk, Kintsch 83]. In this model, text comprehension is simulated by using propositions to represent information in the text. The ability of readers to incorporate text information into their understanding is based on a variety of factors, including the coherence of the text and the background knowledge of the reader.

In a linear text, a writer typically makes paragraphs and sections flow from one to the other in a coherent way. This aids the reader in structuring the information in the section to fit with what has been read previously. If there is little coherence between sections and a user is not familiar with the domain, then the user must make bridging inferences. These inferences consume the reader's resources, typically resulting in lower comprehension.

In a hypertext, a reader has the possibility to jump to a variety of text nodes. It may not be possible to maintain good coherence for all possible links, resulting in additional processing load for the reader. The reader must make decisions about what node to jump to next and maintain information for navigating back to the starting node. This additional processing load suggests that hypertext readers might not do as well as readers of linear text. However, it can also be argued that the additional effort that goes into deciding what node to jump to should result in a stronger understanding of the structure and relationships

between pieces of information in the text. Thus, the additional effort may cause a reader to perform additional elaboration of the text.

An experiment was performed that investigated readers' comprehension of a linear or a hypertext version of a chapter from an undergraduate level economics textbook, manipulating the background knowledge of the subjects. The chapter, originally written in the linear form, was converted into a hypertext running in Hypercard with links between related information nodes. An additional version of the hypertext was created that automatically inserted additional specific pieces of text when a subject made an noncoherent jump. This added text was designed so that it would maintain coherence between the nodes for any possible jump. All of the subjects had no previous knowledge of economics and half were initially trained with some economics background knowledge. This gave them a general situation model before reading the text. Half of the subjects were given instructions to find specific pieces of information, while the other half of the subjects read the chapter for general knowledge. After reading the hypertext or linear text, comprehension was measured using a variety of measures to examine both the textbase (e.g. reproduce or paraphrase the information) and the situation model (make inferences or use the information to solve problems).

The measurements were in depth: reading times, answers to questions, and recall protocols. However, in all cases, the results were negative. Neglecting minor variations, readers performed about as well with hypertext, whether it was made coherent or not, as they did with linear text. This conclusion agrees with

Upon closer inspection, the reasons for this finding became apparent: 55% of the time, readers traversed the hypertext top down, left-to-right. They behaved exactly as if it were a linear text. In fact, counting only "incoherent" jumps that do not remain on the same level in text hierarchy and within the same branch of the tree, then the total number of jumps in the hypertext conditions was approximately the same as the number of jumps by skipping pages in the linear text. Thus, under the conditions of the present experiment, readers read the hypertext and the linear text in much the same way, and hence performed similarly.

These results pose the question, under what conditions can differences be expected between readers of hypertext and readers of linear texts? We suggest two possibilities. First, when texts are very long and the domain very complex, a combination of hypertext and information retrieval systems, such as is found in SuperBook [Landauer et al. 92], may prove effective. Second, hypertext may prove useful in tutoring systems that attempt to adjust text properties to demands, skills, and background knowledge of readers [Kintsch et al. 92]. Further research is needed to determine if these expectations are justified.

3.5 Conclusions from Theoretical and Empirical Work

In summary, our research on the use and formation of situation and system models has included building theories (system and situation model distinction, planning routine tasks), performing empirical work (experiments with retrieval systems, protocol analyses of planning), and applying computer simulations of human performance (NETWORK). This theoretical and empirical work provides a conceptual background for the systems discussed in the text section, especially the CODEFINDER and IRMAIL systems. A close integration between psychological research and system building efforts has been achieved. For the future, additional research remains, especially in the area of the acquisition of situation models. The research on learning from texts and hypertext has been experimental and theoretical.

The work performed on this project has stimulated a great deal of activity not directly associated with the project. Peter Foltz has expanded his work on retrieval systems in collaboration with researchers from Bellcore (6 archival publications at this point). The Mannes & Kintsch work on planning has ben taken up by Stephanie Doane (now at the University of Illinois) and applied to problems in using the UNIX system (3 publications so far, and an application to NSF to continue this work). The work on learning from text has been used as the basis for a project supported by the Mellon Foundation for which W. Kintsch is the principal investigator (2 presentations and one technical report are already available on this project). Furthermore, both Evelyn Ferstl and Peter Foltz are basing their dissertations for the Ph.D. degree on the research that was started as part of the present ARI project (see Appendix II).

4. Innovative System Building Efforts

In the systems building effort, we have applied the theorectical work of the situation and system models, focussing on information access issues. Section 4.1 describes the relationship of the situation-system model theory to information retrieval, develops an integrated systems architecture, and provides a scenario for illustration. The remaining sections describe the particular conceptual bases and implementations of system components supporting the architecture. Section 4.2 describes a system called INFOSCOPE that supports users' personalization of Usenet News for retrieving desired information. Section 4.3 discusses CODEFINDER, a system that combines psychological models of strategic memory retrieval with information management techniques for automatic retrieval. Finally, Section 4.4 describes the EXPLAINER system for helping users judge the relevance and determine the applicability of retrieved items by exploring examples. Section 4.5 describes IRMAIL, an experimental system that uses a standard electronic mail interface for retrieving information. Finally, Section 4.5describes IRMAIL, an experimental system, which combines ideas of electronic mail and retrieval.

The overall goal of the system building efforts has been to provide a wholistic solution to the problems of information management based on the insights of the theoretical and experimental results discussed in the previous sections. In order to focus the approach, the domain of software reuse was generally adopted due to its dependency on effective information access and its current relevance in software engineering [Tracz 88; Fischer, Henninger, Redmiles 91b].

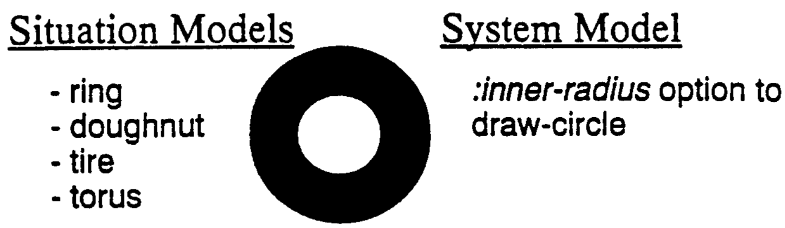
4.1 A Systems Model for Situated Information Access

Situation and System Model Support. When software designers approach a problem, they often begin at a high level of abstraction, conceptualizing the design in terms of the application problem to be solved [Curtis, Krasner, Iscoe 88]. This initial conceptualization must then be translated into terms and abstractions that the computer can understand. The gap between application level and system level in conventional software engineering environments is large. The underlying problem can be characterized as a mismatch between the system model provided by the computing environment and the situation model of the user (see Section 3.1). The same problem has been discussed by Moran [Moran 83] as external-internal task mapping and by Norman [Norman 88] as the gulf of execution and evaluation.

In software design, the situation model is an informal, and often imprecise, representation of what software designers wish to achieve. It includes some understanding of the task to be done: general design criteria, specific components needed, an acquaintance with related problem solutions, etc. In order to develop a solution, users must map their situation models onto terms the system can interpret.

The following simple example illustrates how software reuse can be facilitated by representing knowledge at the level of the situation model. If users wish to draw a ring-like figure, as shown in Figure 4-1, using the software of the SYMBOLICS LISP Machine, they must know the system model which creates this object through the ":inner-radius" option to the "draw-circle" function. The traditional approach to indexing software components is to store them by their name. This representation is specific to the system model because it attends solely to the terms that are important to the system (e.g., how it is called, what options are available). Designers must therefore know to locate this functionality using the name "draw-circle"; i.e., they must be able to conceptualize the problem the same as the SYMBOLICS system.

Our approach is that support systems must contain enough knowledge to assist users in mapping tasks



An informal study we conducted revealed that people conceptualize the task of drawing the object shown with one of the situation models indicated. Indexing for situation models corresponds to *application goals*. If drawing a car, drawing a *tire* would be an example of an application goal. The two system models ("inner-radius:" option to "draw-circle" for the Symbolics Lisp Machine, and blanking out a circular region before shading a circular curve for Disspla) show how system models are indexed by *implementation units*.

Figure 4-1: Situation Models and System Models for a Software Object

conceptualized in their situation model to the system model. The initial suggestion by the system may not exactly fit the user's problem. Mismatches may result from terminology [Furnas et al. 87] or incomplete problem descriptions [Lave 88]. Whatever the cause, a cooperative problem solving process between the system and user is needed to attempt to find an adequate solution.

An Integrated Information Access Model. Our conceptual framework and systems (see Figure 4-2) address these problems as follows. InfoScope (Section 4.2) helps users restructure an information space in order to incorporate into their personal files, information, including software examples that are relevant to their work. It supports personalization through virtual news groups and intelligent, software *agents*. CodeFinder (Section 4.3) uses a combination of two innovative retrieval techniques to support retrieval of software objects without the user having complete knowledge of what is needed. The first technique, retrieval by reformulation, allows users to incrementally construct a query. The second, retrieval by spreading activation, goes beyond inflexible matching algorithms. The combination of these

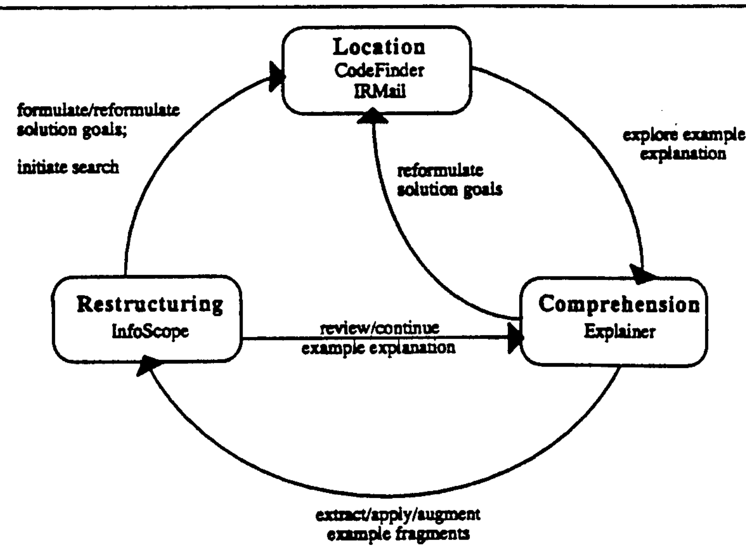


Figure 4-2: An Integrated Information Access Model

techniques yields a flexible retrieval mechanism that provides the means to show the user what components exist and how to access them in the absence of well-formed goals and plans. EXPLAINER (Section 4.4) uses explanations of program examples to help users understand software objects. The examples and explanations help users learn when to apply components and what possible results the components have. Finally, an experimental system, IRMAIL, which combines ideas of electronic mail and location is discussed.

Scenario of Information Access Components in Software Reuse. Our scenario begins with a typical INFOSCOPE user. A user has utilized INFOSCOPE for some time, defining virtual newsgroups and display configurations to match patterns of actual message interest. Over several sessions a newsgroup conversation about graphics environments develops (see Figure 4-4). Eventually a code fragment is posted that generates a circle when it is run. This code fragment is saved by the user from a virtual newsgroup that was previously defined to contain code examples from various discussion groups. From the definition of this newsgroup, the message automatically contains extra keywords such as "graphic," "code," and "example." The user then unconsciously forgets about the existence of this message as new and more interesting topics capture the limited time available for casual information gathering.

At some later point in time, the user wants to draw a "ring" object such as the one shown in Figure 4-1. The integration of CODEFINDER and EXPLAINER demonstrates how the location and comprehension processes of Figure 4-2 work together [Fischer, Henninger, Redmiles 91b]. The designer initially conceptualizes the task in terms of drawing a ring or tire object. The designer begins the process of locating an example by querying CODEFINDER with the "graphics" category shown in the top part of the Query Pane of Figure 4-5. The function "angle-between-angles-p" is retrieved (see Bookmarks of Items Pane). This is not what is being sought, but the description in the Example of the Matching Items Pane provides some retrieval cues, and the designer reformulates the query by specifying the "radius" parameter in the query. This again does not lead to satisfactory results and the designer selects the "Simple Query" command and enters the keywords "circle," "tire," and "ring." This retrieves the "draw-circle" function. After inspecting the description of "draw-circle," the designer decides it is close to what is needed, but lacks the desired feature of specifying the thickness of the line. The designer therefore specifies "draw-circle" to be part of the query, resulting in the query shown in Figure 4-5.

After performing a retrieval, the designer selects "draw-ring" from the Matching Items Pane and decides this function may meet his need. Clicking on the Choose This Button selects the EXPLAINER system and loads the "draw-ring" function for explanation (see Figure 4-8). The designer can explore this example through text, code, and graphic views. The example describes how to create a ring image but offers no suggestion about flattening out the shape and the user returns to CODEFINDER.

Back in CODEFINDER, the designer refines the query by adding the keyword "oval" (see Figure 4-3). Evaluating this new query retrieves the function, "draw-elliptical-ring." The designer again returns to EXPLAINER to review this function and may explore the example until satisfied that this function provides a good basis for his task of drawing a flattened ring.

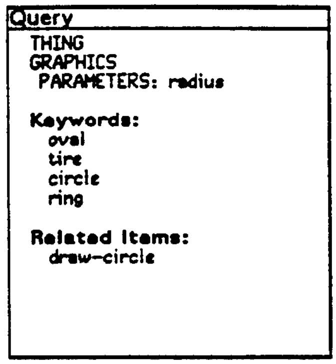


Figure 4-3: Final Query Pane

4.2 InfoScope — Reducing Information Overload through Personalization

While the existence of message distribution systems, such as Usenet News, allows users from distant parts of the world to freely communicate, it also creates difficult problems for users of the systems that access these messages. One problem is the tremendous amount of information overload that can occur when browsing these huge information spaces. This significantly impairs a reader's ability to find interesting information.

Semantic Gap between Situation and System Models. In addition, when accessing news users must constantly perform mappings from their own personal semantics, in which their interest in information is based, to the semantics of a predefined hierarchy of newsgroups. This is further complicated by the fact that the messages are classified by the sender of the message, requiring a guess on the part of the reader as to where someone might classify information on a specific topic of interest (this has been called the vocabulary problem). For example, a reader looking for information about the EMACS text editor on the Macintosh might browse any or all of the newsgroups "comp.emacs," "gnu.emacs," "comp.text.desktop," "comp.editors," or the related Macintosh based newsgroups "comp.sys.mac.misc," "comp.sys.mac.digest," or "comp.sys.mac.apps" (for applications).

The difficulty of this process depends upon the degree of similarity between the semantic interpretation chosen by the message sender and that chosen by the reader. The wider the gap between these interpretations, the more difficult a reader may find this task. Remember, however, that once a message is sent the sender doesn't have to find it but the reader does. This leads to the need for reorganizing the information space based upon the personal semantics of message readers, not message senders (for a more in depth discussion of send time and read time issues see [Fischer, Stevens 91]).

Personalization of an Information Space through Virtual Newsgroups. This research is based on the hypothesis that semantic organization of information should be centered in the semantics of message readers not message senders. Specifically, readers should be able to modify and extend the predefined newsgroup hierarchy with respect to classifications. Extending the previous example, for the duration of users' interest in EMACS they should be able to define a single newsgroup that contains all information about that program. Alternately, a particular user might wish to define a newsgroup that contains only those messages mentioning both EMACS and the Macintosh. Unfortunately, personalization of this type is a difficult process that has been examined in work on adaptable systems and end-user modifiability.

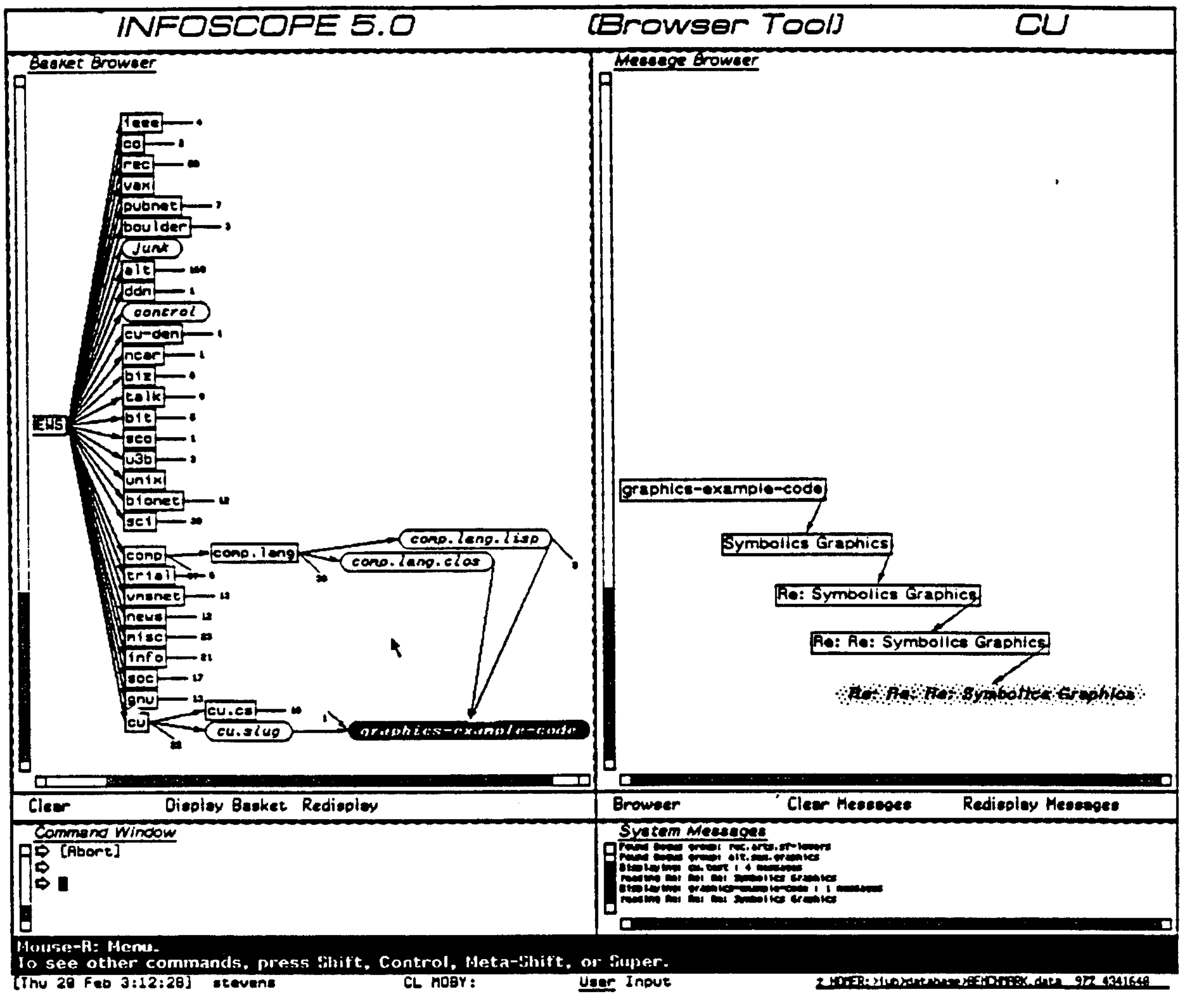


Figure 4-4: Personal, Virtual News Groups in INFOSCOPE

There are four basic stages in the life of a Usenet message. Within each stage, there are several tasks and decisions to be made. A message is created at send time and either addressed to a set of individuals or posted to some sort of bulletin board system. At read time, people looking for information or answering email must decide which of the newsgroups and messages are worth reading. Sometimes, when a message is of particular interest it is stored by the user for later reference. At this point (storage time) the user adds some sort of semantics to the message, ranging from simply specifying a filename to adding keywords and classifications. Lastly, stored messages are retrieved at question time when a question triggers their need.

This difficulty leads to the conclusion that users need help in carrying out personalization. This is especially true for users who are interested in many different topics over long periods of time. Keeping track of these interests as they evolve and disappear may require frequent redefinition of these personalized structures. This creates yet another type of information overload; namely, the overload created by the task of maintaining personal structure. Providing assistance for this process will allow users to concentrate on finding interesting messages.

In order to investigate solutions to these problems the INFOSCOPE system provides tools for defining virtual newsgroups. A virtual newsgroup refines the a priori Usenet hierarchy by filtering out messages regarding specified topics. Filters may currently be defined to include or exclude messages based on the

contents of any header field. This reduces information overload by creating smaller newsgroups containing relatively narrow ranges of coverage. Virtual newsgroups reduce the impact of the vocabulary problem by allowing users to define personalized mappings from keywords to group names. In addition, virtual newsgroups are not limited to a strict hierarchy. By defining filters that search several parent newsgroups, users create a directed graph. A priori newsgroups with similar topics may be combined, and even filtered again to create a totally personal organization. Also, by analyzing the structure added by each user, some of the structure needed by retrieval systems (like HELGON & CODEFINDER) can be added automatically. That structure will not suffer from the same problems as a priori structure since, by definition, it consists of terms known to the user. Finally, that same analysis may yield indications of shifting interest patterns that can be used to help maintain the virtual structure.

Supporting Personalization through Agents. Soon after virtual newsgroups were implemented it became clear, through personal use of this mechanism, that managing the necessary filter definitions could be just as much, if not more effort than manually searching for interesting messages. Newsgroups were originally an answer to the chaotic mess created by so many messages. This led to an a priori structure that is difficult for readers to use. To help with this, virtual newsgroups were implemented which led to a need for managing filters.

INFOSCOPE addresses the filter management problem by implementing agents. These agents monitor user behavior and help with the tasks of creating and maintaining virtual newsgroup filters. In order to make maintenance even easier, agents post suggestions that are based in the user's demonstrated interest patterns. This is possible since the system can analyze past interactions and behavior patterns to determine what virtual structures have been used to fill the gap between a priori and personal semantics. A working hypothesis is that since the suggestions are based on users demonstrated interests, users should understand them. The idea is that by transferring the necessary work to a computer based user model, users will spend less time mapping between different semantics and more time reading interesting messages. Another working hypothesis is that some users will like intrusive agents while others will prefer benign agents. To address this, agents themselves are monitored by supervisory agents. When too many suggestions from an agent are being rejected by the user, that agent is made less intrusive my modifying its tasks or the frequency of its actions. Supervisory agents need not be managed by the user since the user never knows they exist. Regular agents don't need to be managed by users because they have supervisors to do it for them. So, agents transfer the users' task to that of perusing the in formation space and managing suggestions.

Evolution from Usage. One of the big problems in using filters to deal with large information spaces is the effort involved in creating, maintaining, and evolving filters over time. The effort involved can cause enough cognitive strain to make filter management as much trouble as the problems involved in managing large information spaces themselves. Our research investigates these problems in the domain of Usenet News. INFOSCOPE is a news reading system that allows users to create filters that define virtual newsgroups. Virtual newsgroups are extensions to the predefined Usenet hierarchy that correspond directly to specific interests of individual users. In order to address the problems of filter management described above, INFOSCOPE incorporates agents that keep a constantly evolving user model of individual interests. Using various rules and heuristics, agents help users to create and modify their own sets of filters. A significant advantage of this approach is that agents make suggestions that are completed filters. The user is in the position of filter critic instead of filter constructor. This allows users to employ recognition of filter terms instead of recall, and leads to the creation of filters based on the actual reading

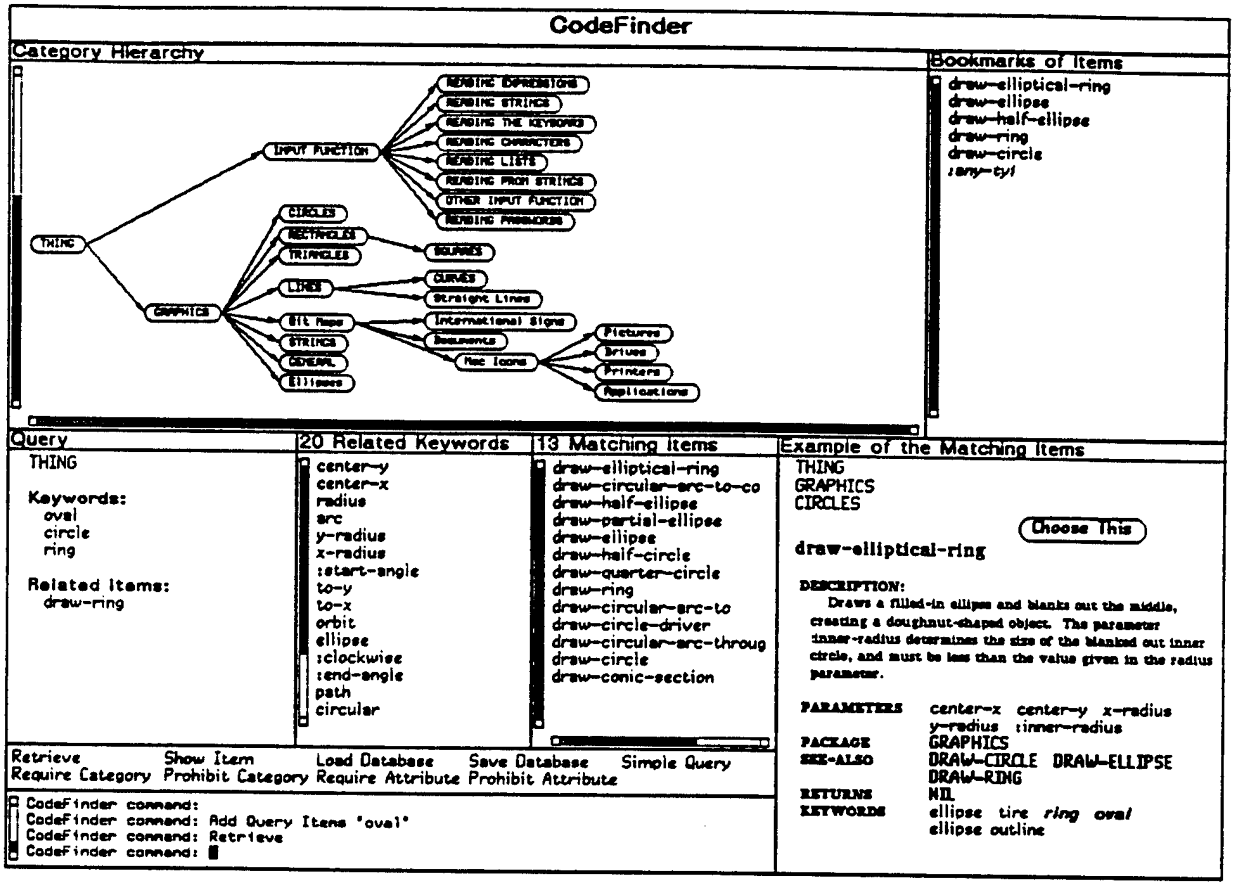


Figure 4-5: CodeFinder User Interface

The CodeFinder user interface is based on Helgon [Fischer, Nieper-Lemke 89]. The **Category Hierarchy** window displays a graphical hierarchy of the information space loaded. In this instance, the information space is a set of graphics functions for the Symbolics Lisp Machine. The **Query** pane shows the current query. The top part of the query specifies two categories (*thing* and *graphics*) and a *parameters* attribute. The bottom part specifies keywords and related items. The query parts combine to retrieve the items in the **Matching Items** pane. The **Example of the Matching Items** pane shows the full entry for an item in the information space. The **Choose This** button loads the example item into Explainer for a detailed explanation. The **Bookmarks** pane holds a history of the objects that have appeared in the **Example of the Matching Items** pane. The **Matching Items** pane shows all items matching the current query, by order of relevance to the query. The **Related Keywords** pane shows keywords retrieved by the query. Any of these keywords can be added to the query through mouse action. The remaining panes allow users to specify commands by mouse action or keyboarding (with command completion).

patterns of individual users. This can be especially helpful in situations where users do not recognize the patterns of interest they exhibit. InfoScope is an operational system running on Macintosh computers.

4.3 CodeFinder — Combining Strategic and Automatic Models of Retrieval

CodeFinder is an extension of Helgon that provides facilities to help users retrieve software objects (see Figure 4-5). CodeFinder combines psychological models of both the strategic and automatic models of memory as discussed in Section 3.1. Strategic processes are supported through tools to incrementally refine a query. Automatic processes are used to generate cues and retrieve information that is used to guide users toward relevant information.

Strategic Retrieval. Retrieval from human memory occurs at different levels of specificity and is in-

crementally refined as new knowledge is retrieved [Norman, Bobrow 79]. If people are not able to retrieve information from their own minds in one try, there is no reason to believe that information retrieval systems can be designed that finds needed information on the first try. The strategic process of refining an information need must therefore be supported. RABBIT [Williams 84] instantiated the paradigm of retrieval by reformulation, based on Norman and Bobrow's (1979) description-based human memory retrieval. Retrieval by reformulation views retrieval as an incremental process of retrieval cue construction. This paradigm creates a cooperative relationship between users and a computer system in which users are given the ability to incrementally improve a query by critiquing the results of previous queries. This incremental refinement allows the formation of stable intermediate query forms upon which users can build until the desired results are obtained. HELGON [Fischer, Nieper-Lemke 89] extended this idea by providing a graphical interface for displaying a concept hierarchy of the information space, and providing facilities for editing by reformulation.

In CODEFINDER, users are given the opportunity to critique a retrieved example to refine a query (see the Example of the Matching Items Pane in Figure 4-5). The quality of the chosen example plays the same crucial role observed in empirical observations of human problem solving [Reeves 91]. The better the example, the easier it is to converge on a satisfactory solution. Therefore, a critical issue in retrieval by reformulation systems is the criteria by which the example is chosen. Previous systems, including HELGON, did not address this issue, but used an arbitrary retrieved item as the example. CODEFINDER enhances the quality of the chosen example by providing a ranking criteria, the activation value, which chooses the item in the information space that is most highly associated with the query.

Automatic Retrieval. Empirical studies of HELGON showed that providing natural means of query formation, which takes advantage of the way human memory works, should lead to better retrieval systems [Foltz, Kintsch 88]. CODEFINDER uses an associative form of spreading activation [Mozer 84; Belew 87; Cohen, Kjeldsen 87] based on a psychological model of human memory [Anderson 83; Kintsch 88] to further enhance the retrieval cues offered by HELGON and other retrieval by reformulation systems.

CODEFINDER uses an associative spreading activation method based on a connectionist relaxation procedure for locating software objects. This technique uses associations to retrieve items that are relevant to a query but do not exactly match it, thus supporting designers when they cannot fully articulate what they need. CODEFINDER uses an extension of Mozer's model [Mozer 84]. It refines the model by noting that the weight on inhibitory links plays a crucial role in retrieval: the higher the inhibitory weights, the less activation will be available to "induce" keywords and documents (see discussion below on induced keywords). Because this inductive process is key to the performance of the model, it would be best to start with a low link weight, then gradually increase the level of inhibition to ensure stability. This is similar to simulated annealing techniques [Bein, Smolenksy 88]. Both the Mozer and the Bein and Smolensky models used a constant link weight between terms and documents; CODEFINDER extends the model further by making use of inverse document frequency measures for link weights. This technique assigns high link weights to terms with high discrimination values (i.e. terms referenced by fewer objects have a higher discrimination value than those referenced by many objects).

Combining Strategic and Automatic Retrieval. A CODEFINDER associative network represents terms (keywords) and software items as nodes in an associative network (see Figure 4-6). Links are weighted, with initial weights determined by an inverse document frequency measure. Activation is spread in the following manner (formal equations of the process can be found in [Mozer 84]). Nodes with positive

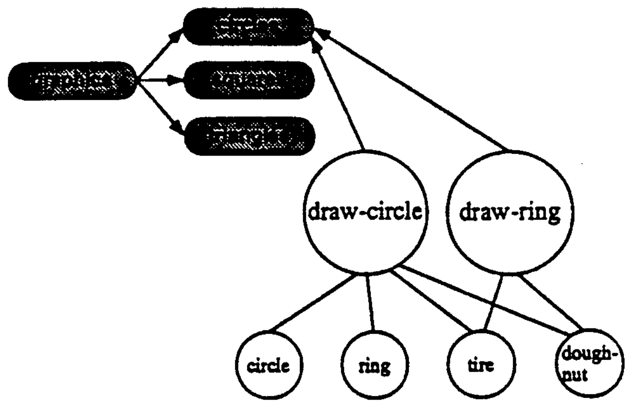


Figure 4-6: Indexing by Application Goals

The indexing architecture of CODEFINDER makes use of both a hierarchical arrangement of categories and an associative index of keywords. In this figure, ovals represent categories, the smaller circles represent keywords, and larger circles are code objects (keywords and code objects together compose the associative network). The function *draw-circle* is divided into two objects, one represents the function as a whole, and the other represents an option to *draw-circle* (*draw-ring*), which draws a ring. A connection between a keyword and code object means that there is an association between the keyword and code object. An arrow from a code object to a category means the object is contained within the category.

---

activation values pass their current activation value to each node they are linked to. Each unit then receives activation values from associated nodes, modified by link weight. For example, if two nodes are connected by a link with a weight of .5, then half of the sending node's activation value will be received. The node computes the sum of received activation values, modulated by fan-in and decay parameters. The resulting value is fed into a squashing function to normalize activation values between boundaries of -.2 and 1, as per [McClelland, Rumelhart 81].

The process starts with a query, which consists of term or document nodes. The query nodes are given an activation value of 1.0, which remains clamped during the spreading activation process. The system is allowed to cycle through the above procedure until stabilization is reached or a maximum number of cycles is reached. Stabilization occurs when a cycle results in small changes to node activation values.

The associative spreading activation model in CODEFINDER allows flexible inferencing and reasoning with incomplete or imprecise information [Mozer 84], which enhances indexing. In most keyword approaches, if a query does not include keywords associated with a particular object, that object will not be retrieved. Figure 4-6 shows that the keywords "ring" and "circle" are not connected to "draw-ring," which draws the desired doughnut-like object. These keywords will activate the "draw-circle" node, which in turn activates keyword nodes "tire" and "doughnut". These keywords will work together to activate the "draw-ring" node, retrieving the proper object. As this example shows, connections between keywords and software objects are *soft constraints*, which allows some flexibility in indexing. These induced keywords compensate for inconsistent indexing because keywords are dynamically related through the items they index. These keywords also provide cues for reformulation. By displaying the induced keywords (see the Related Keywords Pane in Figure 4-5), users are given an idea of the terminology used in the information space, minimizing the chance that a query is constructed with keywords that the system does not 'know' about.

Problems	Support
retrieval by reformulation:	**spreading activation:**
- classification errors	- soft rules
- navigation problems	- augments hierarchical navigation with indexing
- no ranking of retrieved items	- ranks relevance by degree of association to the query
spreading activation:	**retrieval by reformulation:**
- measures relevance by degree of association	- humans choose relevant items from retrieved list
- lacks semantics	- humans assign semantics

Figure 4-7: Complementary Retrieval Paradigms

Many of the problems observed with retrieval by reformulation are addressed by spreading activation techniques, and vice versa. The methods complement each other, resulting in a system that is superior to either in isolation.

Cooperative Problem Solving. Work on CODEFINDER has been motivated by a theory that cooperative dialogues between people and computers can be improved by analyzing the relative strengths of humans and computers for a specific domain [Henninger 90]. For retrieval systems, this analysis must come from psychological theories of human memory. Human knowledge retrieval can be characterized as consisting of two component processes: the construction of a retrieval cue and the action of that cue on memory [Walker, Kintsch 85]. Construction of a retrieval cue is conscious and involves problem solving and control strategies. The execution of cues on memory is automatic and beyond conscious control. Because people are able to derive rich strategies for constructing cues that defy computer simulation [Walker, Kintsch 85; Williams 78], and because people are able to understand what the information can be used for, they should be given the task of directing the search. Computers, which are able to keep track of vast amounts of information but do not know how it should be applied to the problem at hand, can model the unconscious aspects of retrieval.

Retrieval by reformulation can be used to support the strategic aspects of retrieval by presenting examples of retrieved items that can be critiqued. This gives people the ability to assess relevancy and incrementally define a query. Spreading activation and related methods have been used to simulate a variety of psychological results on human memory [Anderson 83; Kintsch 88]. The flexibility of spreading activation relieves users from having to know a great deal about the structure of the information space, allowing them to concentrate on more creative tasks [Henninger 90]. Retrieval by reformulation and spreading activation support this delegation and complement each other by addressing each other's weaknesses (see Figure 4-7).

Extensions to HELGON. The main difference between CODEFINDER and HELGON is that CODEFINDER adds the spreading activation retrieval mechanism to HELGON's subsumption model. HELGON-style queries are still possible within CODEFINDER, but users are also allowed to construct simple keyword queries that circumvent some of the problems users have constructing structured queries and does not suffer from the "no matching items" problems observed in HELGON studies [Foltz, Kintsch 88]. CODEFINDER also ranks its retrieval set by strength of activation values instead of displaying them in alphabetical order. The spreading activation process provides soft constraints that avoids the need for

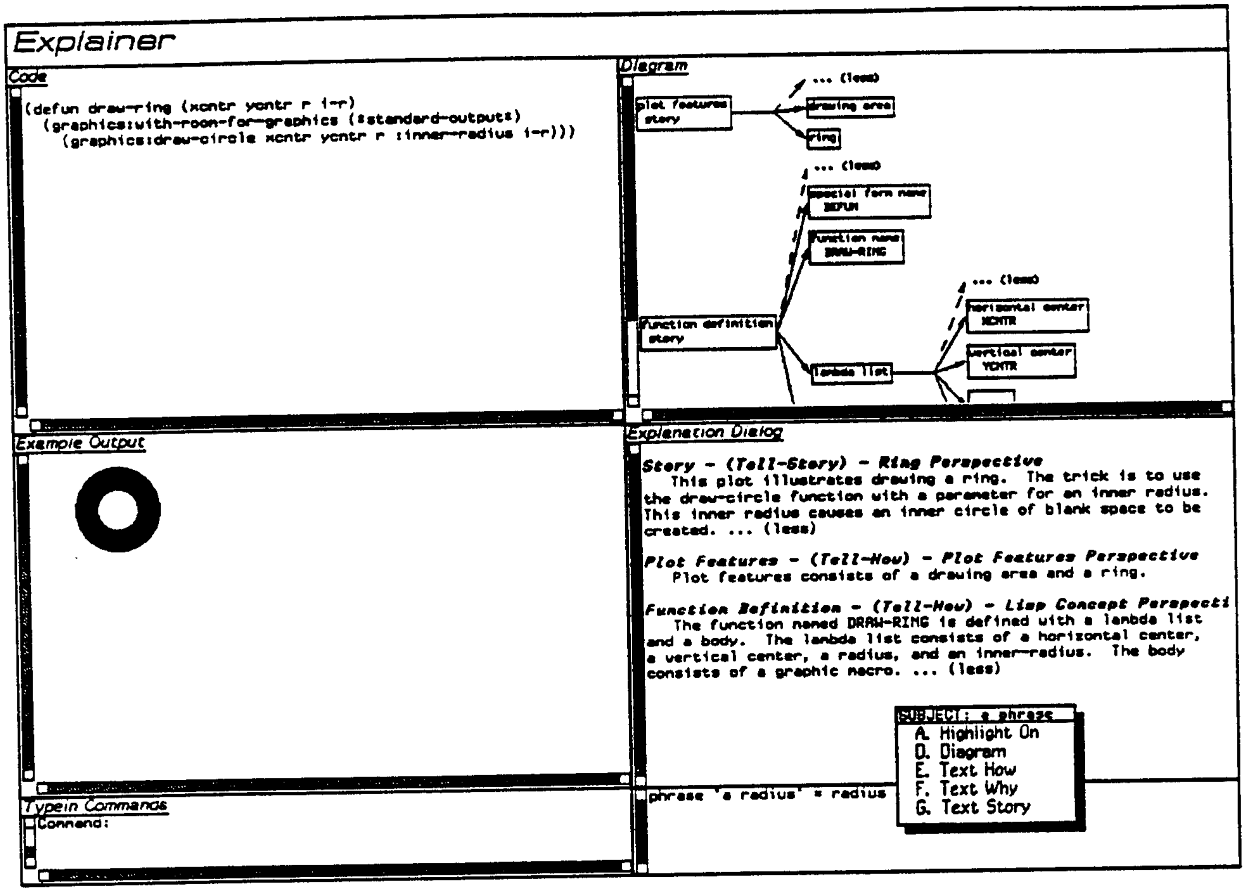


Figure 4-8: Exploring an Example in EXPLAINER

For each example, EXPLAINER begins by displaying the code and sample execution on the left half of the screen. An initial diagram of features and text description are displayed on the right. A pane at the bottom right gives quick descriptions of screen objects the user pauses on with the mouse. The pane at the lower left echos menu commands.

Users ask questions, such as "how" or "why," by clicking on a piece of the text, code, diagram, or graph. A menu appears, and the users choose a question for the selected item. The pop-up menu shown here was initiated by clicking over the phrase "a radius" in the text pane. Other actions include highlighting relations between the different views of an example and expanding the diagram. This screen reflects question-answer history already in progress.

---

users to intimately know the structure of the information space to construct a query. CODEFINDER also contains enhanced capabilities for editing the information space, adding modified code to the information space, refining the representation of a software object (adding keywords, etc), displaying source code, and other facilities.

Ongoing work in CODEFINDER is concentrating on constructing a large database of extensions to the EMACS editor written in ELISP programming language. This information space will be used to evaluate CODEFINDER's ability to help users find ELISP source code objects, and further investigate the process of software reuse in a location-comprehension-modification fashion.

4.4 EXPLAINER — Judging Relevance and Applicability through Examples

The goal of software reuse is to incorporate design ideas and components (e.g., functions, whole programs, or code fragments) of existing software into a new program [Standish 84]. Many problems frustrate the complete success of software reuse [Fischer 87a; Biggerstaff, Richter 87]. One problem is that before software components can be reused, they must be understood by the potential re-user. The role of program examples in supporting a user's understanding is the focus of the approach described below.

A prototype software tool called EXPLAINER has been developed to help users explore and understand existing program examples. EXPLAINER presents four views of a program example: code, sample execution, diagram of features, and text explanations (see Figure 4-8). Any of these views may serve as a starting point for exploring the example. The domain of EXPLAINER is a set of plotting functions on the SYMBOLICS LISP-machine, approximately 60 functions. Example programs collected into a catalog demonstrate different graphic features supported by the functions. A user would be able to select an example from the catalog and get explanations about how the example implements a certain feature, such as drawing a line or curve. The intended users of EXPLAINER are LISP programmers who have some familiarity with graphics concepts but are not experts with the specific functions on the SYMBOLICS.

The purpose of the EXPLAINER research is twofold. First, the EXPLAINER program tool provides a specific framework for observing how people make use of examples when solving new programming tasks. Second, it provides a test bed for techniques of representing knowledge in program examples.

Cooperative Problem Solving. The process of programming can be interpreted more generally as a design process [Fischer 89]. When two people, or a person and a computer, are involved, the process is characterized as cooperative design: both parties bring their own strengths to solving the task at hand. Users bring their ability to understand: i.e., an initial understanding of a problem task and their ability to generalize, draw analogies, and modify their understanding of the task. Through EXPLAINER, the computer brings to bear a catalog of examples along with a representation for explaining program ideas through text, code, diagram, and graphics.

In a design session, the cooperative "understanding" that is developing is the building of a bridge between the situation model and the system model. The situation model is the way a person relates facts or ideas to what they already know. In the context of software reuse, the situation model may be viewed as the user's understanding of a problem and how it might be solved. This situation model resides and evolves in the mind of the user. The system model is the way a computing system is structured to allow a solution to be implemented, namely, the functions that could be combined to program a solution. In the context of software reuse, these functions and partial or whole solutions are demonstrated in existing program examples. The system model resides in the representation of these examples. EXPLAINER helps its users *reformulate* their problem in terms of the system model by demonstrating with examples how similar problems were already solved.

Examples have a practical use in constraining design solutions in high-functionality [Fischer 87b] programming environments. Two situations arise. One is characterized by diversity: different subsystems exist to fill different needs. The system model of how circle and square drawing routines are organized may have little to do with the model of how such drawings are made sensitive for selection through a pointing device. Many system models may be involved in programming one task. In situations

where there are different models at work, examples are a practical means of explanation. An example can simply state: "this is how you do that." Another situation is characterized by redundancy: there may be many ways of doing one thing. In such environments, examples can constrain the possibilities: an example shows *one way* to do something.

Relevance to Learning and Problem Solving. Lewis and Olson [Lewis, Olson 87] propose that the productivity of casual programmers can be increased by adopting a development style that emphasizes the use of example code. This is based on the observation that "when grappling with new material, learners often try to adapt examples of earlier material to suit the present situation, modifying the example by analogy [ibid, p. 9]."

Examples can also help a person's understanding by providing specific situations. Kintsch and Greeno noted that a problem set in the context of a familiar situation helped children solve word arithmetic problems [Kintsch, Greeno 85; Ferstl, Kintsch 92]. The children had a greater basis for relating the problem statement to what they knew and then identifying aspects of the problem. In a similar way, a program example takes a function or subroutine out of a theoretical space and puts it in a context which users have a greater chance of relating to.

A Case-based Systems Approach. The implementation of EXPLAINER implements knowledge about example programs on a case basis [Riesbeck, Schank 89]. The philosophy is that the only knowledge in the system is what can be derived from the known examples. The more varied the collection of examples, the broader the scope of knowledge. The *knowledge* per se is a semantic net of concepts. The concepts are represented by CLOS classes. In keeping with the case-based approach, the connections in the semantic net reflect their relation in an actual program example. The example program in Figure 4-8 reserves a screen area for the plot. Within that graphics area, the ring is drawn. Consequently, there are CLOS classes corresponding to graphics area and ring.

Perspectives in knowledge representation. Concepts are partitioned into different classification schemes called *perspectives*. The example program in Figure 4-8 was conceptualized as drawing a "ring." The ring in the problem-domain perspective also corresponds to a function call in the LISP programming language perspective. Thus one element of a program example is related to concepts in many perspectives. See also Figure 4-9

Currently, examples are entered into EXPLAINER by running a parser that digests LISP code into a network of concepts in the LISP programming language perspective. However, the links to concepts in other perspectives are done by hand.

Evaluating the Approach. EXPLAINER and its example-based approach in general have been evaluated with two informal experiments. In each, the subjects had similar characteristics: familiarity with LISP but not with the domain of the examples, computer graphics.

The purpose of the first experiment was to observe in general how programmers make use of examples in solving new tasks, and in particular, what knowledge would ideally be represented in the EXPLAINER system. Subjects were given a programming task and an example program they were told was related to one possible solution of the task. Instead of using the EXPLAINER system, they had a human consultant available for answering any questions they had. The goal was to observe the widest spectrum of knowledge needed in a cooperative reuse dialog. An encouraging preliminary result was that subjects

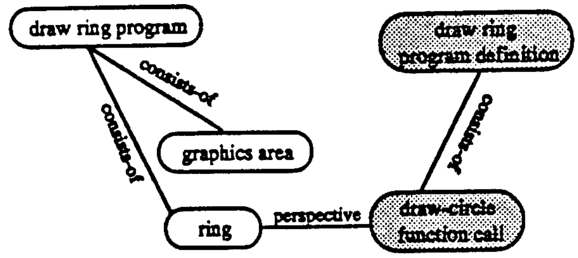


Figure 4-9: Representing Examples through Perspective Mappings

Examples are represented by networks of their constituent concepts and are organized into different perspectives. EXPLAINER relies on the mapping of concepts between perspectives. Here is shown a mapping between concepts in a plot perspective (clear ovals) and a LISP programming language perspective (shaded ovals).

---

were able to make analogies between their assigned tasks and the tasks illustrated by the supplied examples, and then adapting the examples to new solutions An unanticipated observation was that subjects did not articulate every question they had. To address this problem, a method of volunteering information in EXPLAINER by tracking the mouse was added and more training in the approach was given in the next experiment.

The second experiment tested the actual EXPLAINER implementation. In particular, it tested whether the limited set of knowledge and question types were sufficient for understanding an example. The evaluation took place in the context of an advanced LISP programming class. Students were given an example to critique and answer questions about. Of twelve students in the class, half used the EXPLAINER tool and half used other methods such as reading source code and brief accompanying documentation. Encouraging results were that the users users scored higher than non-users with less time spent working overall. Furthermore, the greater the percentage of their time users spent with the tool, the less total time it took to answer the questions.

Future plans include more rigorous testing and perhaps incorporating the EXPLAINER system into the the LISP course on an on-going basis.

4.5 IRMAIL — Providing Information Access with Minimal Interface

IRMAIL [Foltz 92] is a experimental retrieval system developed in order to investigate issues in developing a high functionality retrieval system while minimizing problems of information access. The goal of the system was to allow people to access information while using a familiar user interface, namely their own electronic mail system. The system permits users to mail queries to a central knowledge source which then sends return mail with the items that best match their queries. Thus, users need not learn any particular interface to the retrieval system, but must just be familiar with how to send mail.

Coupled with the mail-based interface is a powerful retrieval engine using Bellcore's Latent Semantic Indexing (LSI) [Deerwester 90]. LSI permits retrieval of textual information based on the semantic content of the words by constructing a semantic space more appropriate for information retrieval. Thus, a query on "ergonomics" will return articles that just use the words "human factors" since the terms are semantically related. Research on LSI [Deerwester 90; Dumais et al. 88] shows that retrieval of relevant docu-

ments is significantly improved compared to direct keyword matching. LSI therefore works to structure the information space in a manner to produce effective retrieval.

Users' retrieval strategies are also supported in IRMAIL through the use of relevance feedback. Given items returned from a query, users can indicate the relevancy of items in order to perform a new query which will return a greater number of relevant items. This reduces the load on the users permitting them to concentrate more on their task at hand rather than at trying to find information relevant to their task.

Overall, IRMAIL works as a testbed for developing simple interfaces to complex information stores. Through providing automatic structuring of the information space and support for users' retrieval strategies it permits testing of the minimum type of interface necessary in order to interact with such as system. Currently an implementation of IRMAIL is in use on a database of HCI related articles and has received wide use by the HCI community.

5. Future Research Issues

In an information n rich society, the resource in short demand is not information but human time and willingness to attend to and retrieve relevant information [Simon 81]. In this context, our research efforts address some of the most pressing problems: (1) increasing the usability of high functionality systems (without decreasing their usefulness); (2) supporting reuse and redesign; (3) assisting users in finding the relevant information in complex, poorly structured information stores; (4) decreasing the information overload problem (5) creating a shared understanding of an information space, and (6) supporting new learning strategies such as learning on demand and combining demand-driven techniques with training. These issues will be briefly discussed below.

Increasing the usability of high functionality systems (without decreasing their usefulness). The research efforts described in this project demonstrate that specialized support is necessary to make complex systems and information spaces usable and useful. A central theme in this support is the proper distribution of tasks between user and computer. It is crucial that each participant in this interaction be responsible for the tasks best suited to a specific set of capabilities. We have demonstrated that the organization of information spaces is a complex task and that users benefit greatly by having system assistance in doing it. However, it is important to balance that with the individualized aspects of personal information stores and retrieval techniques.

In order to increase the usability of high functionality systems even further it will be necessary to personalize not only the organization and presentation of information, but also the way in which these systems are explored and learned. The recognition of the fact that tasks take place in the context of situated problem solving activities means that the knowledge required by individual users will vary depending upon the specific task. By supporting personalized learning, systems can grow in complexity without sacrificing the usefulness that makes them valuable.

Supporting reuse and redesign. Our work to date has shown the value of showing programmers when and how to apply and combine examples. We have shown that the existence of libraries of components is insufficient for successful reuse. Designers must be supported in the process of locating relevant examples and understanding what the example does and how. Our information spaces to date have been rather small, but we feel our techniques and support tools will scale to larger software repositories. To evaluate this intuition we have been applying our location tool to the domain of EMACS customization, where thousands of ELISP functions have been developed by a number of authors. This effort will provide additional insights on how our work can be applied to large-scale software engineering projects.

While our approach has been concerned with the downstream software engineering activities, such as code development, we are intimately aware of the current void in tools to support upstream activities such as requirements definition and design specification. Some work has been accomplished in the domain of kitchen design [Fischer, Nakakoji 91], and we are interested in applying and refining these techniques for other software design domains.

Assisting users to find the relevant information in complex, poorly structured information stores. While our approach has concentrated on the user-centered perspective of trying to find information, another possible perspective realizes the potential for the computer to act as an advisory agent [Hill, Miller 88], or a critic [Fischer et al. 91], identifying potentially relevant information and displaying it to the user. In this perspective users do not realize that useful information exists, and may not know of other

means of performing a task. Users may therefore not be motivated to look for information. The system can identify users' information needs by recognizing sub-optimal behavior and identify an information space that the user may want to become aware of. Querying and browsing can then proceed if the system misjudges the information need.

Information needs and resulting queries or browsing behavior always arise from the larger context of a problem that needs solving. Queries alone often represent a decontextualized information need. In principle context can be explicitly stated in the query, but this would greatly complicate the query. Another approach would be to use the partially completed design artifact to partially represent the context. For example, let's say that a designer has partially completed an E-mail system and is currently designing the header fields. A query including the terms "headers" and "field" would retrieve a great deal of information from such domains as news readers or database records. Including context information in the query would better anticipate the user's need for E-mail header fields. This can be done with a background query which uses a representation derived from the partial E-mail system design and a specification component to retrieve useful information. The representation and use of background queries are major technical issues that need to be further investigated.

The research on situation models remains too abstract to be of immediate use in system design. Further research is needed to investigate the structure and contents of situation models to improve indexing and explanation methods. The CODEFINDER work has investigated how indexing by application goals can enhance the retrievability of information. Further investigation of typical situation models, what they look like and how they are used, as well as how they can be acquired for system use, will provide better theories on how retrieval systems can be improved. Explanations can be better tailored to use situations through an enhanced understanding of how people think and reason about examples.

We have argued that in complex information domains, retrieval of examples is only a partial answer to the information overload problem as people will have difficulty understanding the examples. Location and comprehension should not be looked at in isolation of each other. Further investigation and observation of the intertwining of location and comprehension are needed to better understand how these processes can be supported with computer-based tools.

Decreasing the information overload problem. Many people living in our society have to cope with a tremendous amount of information [Norman 93], We have to take into account not only the producers of that information but also its consumers. Producing more information will not make computers helpful; instead, systems that help us attend to the most useful, most interesting, or most valuable information are needed [Simon 81; Fischer, Stevens 91]. We have developed systems that reduce information overload by helping users to reduce the information space to manageable and recognizable chunks. This is accomplished by, (1) allowing users to make more efficient use of their time, (2) giving users fewer tasks to perform, (3) making more time available to attend to interesting information, (4) allowing users to reorganize information spaces and incrementally define queries, and (5) providing the user with appropriately contextualized assistance.

The problems of information overload are varied and complex. There are at least as many proposed solutions as there are problems to solve, but none of them are sufficient. The global village is growing fast as more and more computers are linked to the worldwide Internet. The problems presented by an information space the size of Usenet news, or a specific problem domain will seem small compared to the

search for information in the networks of the future. Therefore, the scalability of our information access systems is a major avenue of research for the future. In addition, there will be many new types of information available including video, animations, simulations, virtual realities, services currently available only through the use of telephones or physical transportation, and in general complex objects that make text based methods of retrieval and organization obsolete. Rather than reduce the information overload problem by making information spaces seem smaller than they actually are, techniques will have to be developed that fundamentally change how information is conceived and dealt with.

Creating a shared understanding of an information space. It has been empirically observed that person-to-person communication is best understood as a process in which a mutual understanding is evoked [Suchman 87;Reeves1991]. We have begun to investigate how these kind of communicative processes can be brought to the domain of human-computer interaction for finding information in complex domains. The combination of spreading activation and retrieval by reformulation results in a cooperative situation where the computer displays associations in its information space that the user can apply to refine a query. The ability to choose information from different perspectives provides a context sensitive means to gain access to information.

At this point in time, the computer has acted in a very passive manner to the issue of mutual understanding. The techniques we have used do not actively try to model the user or otherwise attempt to understand the problem the user is trying to solve. User modelling techniques have been traditionally been employed to this end. Another possibility is to model and categorize past problems that a system has been used to solve. The system can then attempt to match a new problem solving episode to a previous one and use that information to suggest possibilities to the user. To the extent that the problem has been attempted before and the system is able to find that match, a form of mutual understanding will be achieved.

People create shared understandings among each other through language. While the field of natural language processing continue to mature, we feel the contribution of this field will be limited to the extent that the issue of mutual understanding and context are ignored. Psychological theories of discourse comprehension have begun to address how understanding arises through an integration of long-term memory in the context of a discourse [Kintsch 88]. Further research is needed to better understand this process the extent that the resulting models can be employed in systems to improve their ability to "understand" what the user wants and needs.

Supporting new learning strategies such as learning on demand combining demand-driven techniques with training. The methods that have been proposed and researched within the context of this grant have focused on support for bridging the gap between the situation and system model within the context of a specific problem. We have demonstrated that these techniques have a significant positive effect in HFCS where the complexity of the system exceeds the human capacity to understand the system completely. But the reliance on learning about a system from individual cases alone may in some cases be inefficient or lead to suboptimal interaction with the system. For example, if one learns that the ":inner-radius" option to "draw-circle" can be used to draw a ring, they may not be able to transfer this knowledge to "draw-ellipse" without understanding the principles and generality associated with the ":inner-radius" option. Studies in the transfer of knowledge have shown that people are often unable to use the methods of one kind of problem solving to an analogous, but different, situation [Lave 88; Gick, Holyoak 80]. Because of this, training must often be given within the relevant situation or explicit ex-

amples provided of how the concepts can transfer to a new situation.

It is therefore perceived that a need exists to integrate training methods with the demand-driven methods we have proposed. In addition to providing a mapping between the situation and system models, we need to detect situations where some training material can teach the user the broader context in which a concept can be applied. This would differ from traditional training methods in that up-front training is minimalized in favor of teaching concepts in the context in which they can be used. A greater understanding of a user's model of a situation can help detect what background context must be provided in order for the training methods to be effective.

References

[Anderson 83]
J.R. Anderson, *The Architecture of Cognition*, Harvard University Press, Cambridge, MA, 1983.

[Bein, Smolenksy 88]
J. Bein, P. Smolenksy, *Application of the Interactive Activation Model to Document Retrieval*, International Workshop on Neural Networks and their Applications, Nimes, France, 1988, (also published as Technical Report CCU-CS-405-88, Dept. of Computer Science, Univ. of Colorado-Boulder).

[Belew 87]
R.K. Belew, *Adaptive Information Retrieval: Machine Learning in Associative Networks*, Technical Report 4, Cognitive Science and Machine Intelligence Laboratory, University of Michigan, 1987.

[Belkin, Croft 87]
N.J. Belkin, W.B. Croft, *Retrieval Techniques*, Annual Review of Information Science and Technology (ARIST), Vol. 22, 1987, pp. 109-145.

[Biggerstaff, Richter 87]
T.J. Biggerstaff, C. Richter, *Reusability Framework, Assessment, and Directions*, IEEE Software, Vol. 4, No. 2, March 1987, pp. 41-49.

[Bookstein, Swanson 75]
A. Bookstein, D.R. Swanson., *Probabilistic Models for Automatic Indexing*, Journal of the American Society for Information Science, Vol. 26, No. , 1975, pp. 45-50.

[Bovair, Kieras 85]
S. Bovair, D.E. Kieras, *A Guide to Propositional Analysis for Research on Technical Prose*, in B.K. Britton, J.B. Black (eds.), *Understanding Expository Text*, Erlbaum, Hillsdale, NJ, 1985.

[Bush 45]
V. Bush, *As we May Think*, Atlantic Monthly, Vol. 176, No. 7, July 1945, pp. 101-108.

[Cohen, Kjeldsen 87]
P.R. Cohen, R. Kjeldsen, *Information Retrieval by Constrained Spreading Activation in Semantic Networks*, Information Processing and Management, Vol. 23, No. 4, 1987, pp. 255-268.

[Curtis, Krasner, Iscoe 88]
B. Curtis, H. Krasner, N. Iscoe, *A Field Study of the Software Design Process for Large Systems*, Communications of the ACM, Vol. 31, No. 11, November 1988, pp. 1268-1287.

[Deerwester 90]
S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, L. Beck, *Indexing by Latent Semantic Analysis*, Journal of the American Society for Information Science, Vol. 6, No. 41, 1990, pp. 391-407.

[Dijk, Kintsch 83]
T.A. van Dijk, W. Kintsch, *Strategies of Discourse Comprehension*, Academic Press, New York, 1983.

[Dumais et al. 88]
S.T. Dumais, G.W. Furnas, T.K. Landauer, S. Deerwester, R. Harshman, *Using Latent Semantic Analysis to Improve Access to Textual Information*, Human Factors in Computing Systems, CHI'88 Conference Proceedings (Washington, D.C.), ACM, New York, May 1988, pp. 281-285.

[Engelbart 63]
D.C. Engelbart, *A Conceptual Framework for the Augmentation of Man's Intellect*, in P.W. Howerton, D.C. Weeks (eds.), *Vistas in Information Handling, Volume I*, Spartan Books, Washington, 1963, ch. 1.

[Ferstl 91]
E.C. Ferstl, *Changes in the Knowledge Structure After Reading a Text*, Unpublished Master's Thesis, University of Colorado, Boulder, 1991.

[Ferstl, Kintsch 92]
E. Ferstl, W. Kintsch, *Learning from Text*, 1992, (in preparation).

[Fischer 87a]
G. Fischer, *Cognitive View of Reuse and Redesign*, IEEE Software, Special Issue on Reusability, Vol. 4, No. 4, July 1987, pp. 60-72.

[Fischer 87b]
G. Fischer, *A Critic for LISP*, Proceedings of the 10th International Joint Conference on Artificial Intelligence (Milan, Italy), J. McDermott (ed.), Morgan Kaufmann Publishers, Los Altos, CA, August 1987, pp. 177-184.

[Fischer 89]
G. Fischer, *Human-Computer Interaction Software: Lessons Learned, Challenges Ahead*, IEEE Software, Vol. 6, No. 1, January 1989, pp. 44-52.

[Fischer et al. 89]
G. Fischer, P.W. Foltz, W. Kintsch, H. Nieper-Lemke, C. Stevens, *Personal Information Systems and Models of Human Memory*, Technical Report, Department of Computer Science, University of Colorado, Boulder, CO, 1989.

[Fischer et al. 91]
G. Fischer, A.C. Lemke, T. Mastaglio, A. Morch, *The Role of Critiquing in Cooperative Problem Solving*, ACM Transactions on Information Systems, Vol. 9, No. 2, 1991, pp. 123-151.

[Fischer, Henninger, Redmiles 91a]
G. Fischer, S.R. Henninger, D.F. Redmiles, *Intertwining Query Construction and Relevance Evaluation*, Human Factors in Computing Systems, CHI'91 Conference Proceedings (New Orleans, LA), ACM, New York, 1991, pp. 55-62.

[Fischer, Henninger, Redmiles 91b]
G. Fischer, S.R. Henninger, D.F. Redmiles, *Cognitive Tools for Locating and Comprehending Software Objects for Reuse*, Thirteenth International Conference on Software Engineering (Austin, TX), IEEE Computer Society Press, ACM, IEEE, Los Alamitos, CA, 1991, pp. 318-328.

[Fischer, Nakakoji 91]
G. Fischer, K. Nakakoji, *Making Design Objects Relevant to the Task at Hand*, Proceedings of AAAI-91, Ninth National Conference on Artificial Intelligence, AAAI Press/The MIT Press, Cambridge, MA, 1991, pp. 67-73.

[Fischer, Nieper 87]
G. Fischer, H. Nieper (eds.), *Personalized Intelligent Information Systems, Workshop Report (Breckenridge, CO)*, Institute of Cognitive Science, University of Colorado, Boulder, CO, Technical Report, No. 87-9, 1987.

[Fischer, Nieper-Lemke 89]
G. Fischer, H. Nieper-Lemke, *HELGON: Extending the Retrieval by Reformulation Paradigm*, Human Factors in Computing Systems, CHI'89 Conference Proceedings (Austin, TX), ACM, New York, May 1989, pp. 357-362.

[Fischer, Stevens 91]
G. Fischer, C. Stevens, *Information Access in Complex, Poorly Structured Information Spaces*, Human Factors in Computing Systems, CHI'91 Conference Proceedings (New Orleans, LA), ACM, 1991, pp. 63-70.

[Foltz 91a]
P.W. Foltz, *A Text Comprehension Model of Hypertext: A theory Based Approach to Design and Evaluation*, SIGCHI Bulletin, Vol. 23, No. 4, October 1991, pp. 70.

[Foltz 91b]
P.W. Foltz, *Human Memory Retrieval and Computer Information Retrieval: What can the two fields learn from each other?*, Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society (Chicago, IL), Lawrence Erlbaum Associates, Hillsdale, NJ, August 1991, pp. 703-707.

[Foltz 92]
P.W. Foltz, *IRMail: A minimal interface to a retrieval system*, Poster presented at CHI 92, May 1992.

[Foltz, Kintsch 88]
P.W. Foltz, W. Kintsch, *An Empirical Study of Retrieval by Reformulation on HELGON*, in A.A. Turner (ed.), *Mental Models and User-Centered Design, Workshop Report (Breckenridge, CO)*, Institute of Cognitive Science, University of Colorado (Technical Report 88-9), Boulder, CO, 1988, pp. 9-14.

[Furnas 86]
G.W. Furnas, *Generalized Fisheye Views*, Human Factors in Computing Systems, CHI'86 Conference Proceedings (Boston, MA), ACM, New York, April 1986, pp. 16-23.

[Furnas et al. 83]
G.W. Furnas, T.K. Landauer, L.M. Gomez, S.T. Dumais, *Statistical Semantics: Analysis of the potential performance of key-word information systems*, Bell Systems Technical Journal, Vol. 62, No. 6, 1983, pp. 1753-1806.

[Furnas et al. 87]
G.W. Furnas, T.K. Landauer, L.M. Gomez, S.T. Dumais, *The Vocabulary Problem in Human-System Communication*, Communications of the ACM, Vol. 30, No. 11, November 1987, pp. 964-971.

[Gick, Holyoak 80]
M.L. Gick, K.J. Holyoak, *Analogical Problem Solving*, Cognitive Psychology, Vol. 12, 1980, pp. 306-355.

[Henninger 90]
S. Henninger, *Defining the Roles of Humans and Computers in Cooperative Problem-Solving Systems for Information Retrieval*, Working Notes of the AAAI Spring Symposium Workshop on Knowledge-Based Human Computer Communication, AAAI, Menlo Park, CA, March 1990, pp. 46-51.

[Henninger 91]
S. Henninger, *Retrieving Software Objects in an Example-Based Programming Environment*, Proceedings SIGIR '91, Chicago, IL, October 1991, pp. 251-260.

[Hill, Miller 88]
W.C. Hill, J.R. Miller, *Justified Advice: A Semi-Naturalistic Study of Advisory Strategies*, Human Factors in Computing Systems, CHI'88 Conference Proceedings (Washington, DC), ACM, New York, May 1988.

[Hintzman 86]
D. Hintzman, *"Schema abstraction" in a Multiple-Trace Memory Model*, Psychological Review, Vol. 93, 1986, pp. 411-428.

[Jones 86]
W.P. Jones, *The Memory Extender Personal Filing System*, Human Factors in Computing Systems, CHI'86 Conference Proceedings (Boston, MA), ACM, New York, April 1986, pp. 298-305.

[Kintsch 88]
W. Kintsch, *The Role of Knowledge in Discourse Comprehension: A Construction-Integration Model*, Psychological Review, Vol. 95, 1988, pp. 163-182.

[Kintsch 92]
W. Kintsch, *A Cognitive Architecture for Comprehension*, in H.L. Pick, Jr., P. van den Broek, & D.C. Knill (eds.), *Cognition: Conceptual and Methodological Issues*, American Psychological Association, Washington, D.C., 1992, pp. 143-164.

[Kintsch et al. 92]
E. Kintsch, D.S. McNamara, N. Songer, W. Kintsch, *Revising the Coherence of Science Texts to Improve Comprehension and Learning I: Traits of mammals*, Technical Report 92-03, Institute of Cognitive Science, University of Colorado, Boulder, CO, 1992.

[Kintsch, Dijk 78]
W. Kintsch T.A. van Dijk, *Toward a Model of Text Comprehension and Production*, Psychological Review, Vol. 85, 1978, pp. 363-394.

[Kintsch, Greeno 85]
W. Kintsch, J.G. Greeno, *Understanding and Solving Word Arithmetic Problems*, Psychological Review, Vol. 92, 1985, pp. 109-129.

[Landauer 86]
T. K. Landauer, *How Much Do People Remember? Some estimates of the quantity of learned information in long-term memory*, Cognitive Science, Vol. 10, 1986, pp. 477-493.

[Landauer et al. 92]
Landauer, T., Egan, D., Remde, J., Lesk, M., Lochbaum, C., & Ketchum, D., *Enhancing the Usability of Text Through Computer Delivery and Formative Evaluation: The SuperBook Project*, in A. Dillon, C. McKnight (eds.), *Hypertext: A Psychological Perspective*, , , 1992, (in press).

[Lave 88]
J. Lave, *Cognition in Practice*, Cambridge University Press, Cambridge, UK, 1988.

[Lewis, Olson 87]
C.H. Lewis, G.M. Olson, *Can the Principles of Cognition Lower the Barriers of Programming?*, in G.M. Olson, E. Soloway, S. Sheppard (eds.), *Empirical Studies of Programmers (Vol. 2)*, Ablex Publishing Corporation, Lawrence Erlbaum Associates, Norwood, NJ, 1987, pp. 248-263.

[Mannes, Kintsch 91]
S. Mannes, W. Kintsch, *Routine Computing Tasks: Planning as Understanding*, Cognitive Science, Vol. 3, No. 15, 1991, pp. 305-342, also published as Technical Report No. 89-8, Institute of Cognitive Science, University of Colorado, Boulder, CO.

[McClelland, Rumelhart 81]
J.L. McClelland, D.E. Rumelhart, *An Interactive Activation Model of Context Effects in Letter Perception: Part 1: An Account of Basic Findings*, Psychological Review, Vol. 88, No. 5, 1981, pp. 375-407.

[Moran 83]
T.P. Moran, *Getting into a System: External-Internal Task Mapping Analysis*, Human Factors in Computing Systems, CHI'83 Conference Proceedings (Boston, MA), ACM, New York, December 1983, pp. 45-49.

[Mozer 84]
M.C. Mozer, *Inductive Information Retrieval Using Parallel Distributed Computation*, ICS Report 8406, Institute for Cognitive Science, University of California, San Diego, La Jolla, CA, June 1984.

[Murdock 83]
B.B. Murdock, *A Distributed Memory Model for Serial-Order Information*, Psychological Review, Vol. 90, 1983, pp. 316-338.

[Norman 88]
D.A. Norman, *The Psychology of Everyday Things*, Basic Books, New York, 1988.

[Norman 93]
D.A. Norman, *Things That Make Us Smart*, Addison-Wesley Publishing Company, Reading, MA, 1993, Expected publication, early 1993.

[Norman, Bobrow 79]
D.A. Norman, D.G. Bobrow, *Descriptions: An Intermediate Stage in Memory Retrieval*, Cognitive Psychology, Vol. 11, 1979, pp. 107-123.

[Norman, Draper 86]
D.A. Norman, S.W. Draper (eds.), *User Centered System Design, New Perspectives on Human-Computer Interaction*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1986.

[Oddy 77]
R.N. Oddy, *Information Retrieval Through Man-Machine Dialogue*, Journal of Documentation, Vol. 33, No. 1, March 1977, pp. 1-14.

[Raaijmaker, Shiffrin 81]
J.G.W. Raaijmakers, R.M. Shiffrin, *Search of Assocative Memory*, Psychological Review, Vol. 88, 1981, pp. 93-134.

[Ratcliff, McKoon 88]
Ratcliff, R., & McKoon, G, *A Retrieval Theory of Priming Memory*, Psychological Review, Vol. 95, 1988, pp. 385-408.

[Reeves 91]
B.N. Reeves, *Locating the Right Object in a Large Hardware Store -- An Empirical Study of Cooperative Problem Solving among Humans*, Technical Report CU-CS-523-91, Department of Computer Science, University of Colorado, Boulder, CO, 1991.

[Riesbeck, Schank 89]
C. Riesbeck, R.C. Schank, *Inside Case-Based Reasoning*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1989.

[Rose, Belew 89]
D.E. Rose, R.K. Belew, *Legal Information Retrieval: A hybrid approach*, Proceedings of the International Conference on AI & Law, Vancouver, BC, 1989.

[Salton, Buckley 90]
G. Salton, C. Buckley, *Improving Retrieval Performance by Relevance Feedback*, Journal of the American Society for Information Science, Vol. 41, No. 4, 1990, pp. 288-297.

[Salton, McGill 83]
G. Salton, M.J. McGill, *Introduction to Modern Information Retrieval*, McGraw Hill, New York, 1983.

[Simon 81]
H.A. Simon, *The Sciences of the Artificial*, The MIT Press, Cambridge, MA, 1981.

[Standish 84]
T.A. Standish, *An Essay on Software Reuse*, IEEE Transactions on Software Engineering, Vol. SE-10, No. 5, September 1984, pp. 494-497.

[Stanfill, Kahle 86]
C. Stanfill, B. Kahle, *Parallel Free-Text Search on the Connection Machine System*, Communications of the ACM, Vol. 29, No. 12, December 1986, pp. 1229-1239.

[Suchman 87]
L.A. Suchman, *Plans and Situated Actions*, Cambridge University Press, Cambridge, UK, 1987.

[Tracz 88]
W. Tracz (eds.), *Software Reuse: Emerging Technology*, IEEE Computer Society Press, Piscataway, NJ, 1988.

[Turner 88]
A.A. Turner (ed.), *Mental Models and User-Centered Design, Workshop Report (Breckenridge, CO)*, Institute of Cognitive Science, University of Colorado, Boulder, CO, Technical Report, No. 88-9, 1988.

[Turner, Greene 78]
A. Turner, E. Greene, *Construction and Use of a Propositional Text Base*, JSAS Catalogue of Selected Documents in Psychology, Vol. MS 1713, 1978.

[Walker, Kintsch 85]
W.H. Walker, W. Kintsch, *Automatic and Strategic Aspects of Knowledge Retrieval*, Cognitive Science, Vol. 9, 1985, pp. 261-283.

[Williams 78]
M.D. Williams, *The Process of Retrieval from Very Long Term Memory*, Technical Report 7801, Center for Human Information Processing, University of California, San Diego, La Jolla, CA, September 1978.

[Williams 84]
M.D. Williams, *What Makes RABBIT Run?*, International Journal of Man-Machine Studies, Vol. 21, 1984, pp. 333-352.

[Williams et al. 82]
M.D. Williams, F.N. Tou, R. Fikes, A. Henderson, T.W. Malone, *RABBIT: Cognitive Science in Interface Design*, Proceedings of the 4th Annual Conference of the Cognitive Science Society (Ann Arbor, MI), Cognitive Science Society, August 1982, pp. 82-85.

Appendix I. Publication Record

Availability of Publications:

Publications are available in the published literature or by contacting us directly:

Francesca Iovine
Department of Computer Science
University of Colorado
Boulder, CO 80303-0430
phone: 303-492-1592
fax: 303-492-2844
e-mail: iovine@cs.colorado.edu

I.1 Archival Publications

1992

1. G. Fischer, B.N. Reeves, **Beyond Intelligent Interfaces: Exploring, Analyzing and Creating Success Models of Cooperative Problem Solving,** Applied Intelligence, Special Issue Intelligent Interfaces, Vol. 1, 1992, pp. 311-332.

2. G. Fischer, A. Girgensohn, K. Nakakoji, D. Redmiles, **Supporting Software Designers with Integrated, Domain-Oriented Design Environments,** IEEE Transactions on Software Engineering, Special Issue on Knowledge Representation and Reasoning in Software Engineering, 1992, Vol 18, No 6, pp 511-522

3. G. Fischer, K. Nakakoji, **Beyond the Macho Approach of Artificial Intelligence: Empower Human Designers — Do Not Replace Them,** Knowledge-Based Systems Journal, Vol. 5, No. 1, 1992, pp. 15-30.

1991

1. S. Mannes, W. Kintsch, **Routine Computing Tasks: Planning as Understanding, Cognitive Science,** Vol. 3, No. 15, 1991, pp. 305-342.

2. G. Fischer, K. Nakakoji, **Empowering Designers with Integrated Design Environments,** in J. Gero (ed.), Artificial Intelligence in Design'91, Butterworth-Heinemann Ltd, Oxford, England, 1991, pp. 191-209.

3. P.W. Foltz, S.T. Dumais, **An Analysis of Information Filtering Methods for Bellcore Technical Memos,** Proceedings of the Bellcore/BCC Symposium on User Centered Design (Livinston, NJ), Bellcore, Piscataway, New Jersey, November 1991, pp. 80-82.

4. S. Henninger, **Retrieving Software Objects in an Example-Based Programming** Environment, Proceedings SIGIR '91, Chicago, IL, October 1991, pp. 251-260.

5. M. Majidi, D. Redmiles, **A Knowledge-Based Interface to Promote Software** Understanding, Proceedings of the 6th Annual Knowledge-Based Software Engineering (KBSE-91) Conference (Syracuse, NY), IEEE Computer Society Press, Los Alamitos, CA, September 1991, pp. 178-185.

6. G. Fischer, **Supporting Learning on Demand with Design Environments,** Proceedings of the International Conference on the Learning Sciences 1991, Evanston, IL, August 1991, pp. 165-172.

7. G. Fischer, K. Nakakoji, **Making Design Objects Relevant to the Task at Hand,** Proceedings of AAAI-91, Ninth National Conference on Artificial Intelligence, AAAI Press/The MIT Press, Cambridge, MA, 1991, pp. 67-73.

8. G. Fischer, A.C. Lemke, T. Mastaglio, A.I. Morch, **Critics: An Emerging Approach to**

Knowledge-Based Human Computer Interaction, International Journal of Man-Machine Studies, Vol. 35, No. 5, 1991, pp. 695-721.

9. G. Fischer, C. Stevens, **Information Access in Complex, Poorly Structured Information Spaces,** Human Factors in Computing Systems, CHI'91 Conference Proceedings (New Orleans, LA), ACM, 1991, pp. 63-70.
10. G. Fischer, S.R. Henninger, D.F. Redmiles, **Intertwining Query Construction and Relevance Evaluation,** Human Factors in Computing Systems, CHI'91 Conference Proceedings (New Orleans, LA), ACM, New York, 1991, pp. 55-62.
11. G. Fischer, S.R. Henninger, D.F. Redmiles, **Cognitive Tools for Locating and Comprehending Software Objects for Reuse,** Thirteenth International Conference on Software Engineering (Austin, TX), IEEE Computer Society Press, ACM, IEEE, Los Alamitos, CA, 1991, pp. 318-328.
12. P.W. Foltz, **Human Memory Retrieval and Computer Information Retrieval: What can the two fields learn from each other?** Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society, Cognitive Science Society (Chicago, IL), Lawrence Erlbaum Associates, Hillsdale, NJ, 1991, pp. 703-707.
13. P.W. Foltz **A Text Comprehension Model of Hypertext: A theory Based Approach to Design and Evaluation,** SIGCHI Bulletin, 23, 4, 1991, pp. 70.
14. G. Fischer, A.C. Lemke, R. McCall, A. Morch, **Making Argumentation Serve Design,** Human Computer Interaction, Vol. 6, No. 3-4, 1991, pp. 393-419.
15. G. Fischer, A.C. Lemke, T. Mastaglio, A. Morch, **The Role of Critiquing in Cooperative Problem Solving,** ACM Transactions on Information Systems, Vol. 9, No. 2, 1991, pp. 123-151.
16. H.-D. Boecker, G. Fischer, H. Nieper-Lemke, **The Role of Visual Representations in Understanding Software,** in D. Partridge (ed.), Artificial Intelligence and Software Engineering, Ablex Publishing Corporation, Norwood, NJ, 1991, pp. 273-290.
17. G. Fischer, **The Importance of Models in Making Complex Systems Comprehensible,** in D. Ackerman, M. Tauber (eds.), Mental Models and Human Computer Communication: Proceedings of the 8th Interdisciplinary Workshop on Informatics and Psychology (Schaerding, Austria), Elsevier Science, Amsterdam, 1991, pp. 3-36.

1990

1. G. Fischer, A.C. Lemke, R. McCall, **Towards a System Architecture Supporting Contextualized Learning,** Proceedings of AAAI-90, Eighth National Conference on Artificial Intelligence, AAAI Press/The MIT Press, Cambridge, MA, August 1990, pp. 420-425.
2. A.C. Lemke, G. Fischer, **A Cooperative Problem Solving System for User Interface Design,** Proceedings of AAAI-90, Eighth National Conference on Artificial Intelligence, AAAI Press/The MIT Press, Cambridge, MA, August 1990, pp. 479-484.
3. G. Fischer, **Cooperative Knowledge-Based Design Environments for the Design, Use, and Maintenance of Software,** Software Symposium'90 (Kyoto, Japan), June 1990, pp. 2-22.
4. G. Fischer, A. Girgensohn, **End-User Modifiability in Design Environments, Human Factors in Computing Systems,** CHI'90 Conference Proceedings (Seattle, WA), ACM, New York, April 1990, pp. 183-191.
5. G. Fischer, A.C. Lemke, T. Mastaglio, A. Morch, **Using Critics to Empower Users,** Human Factors in Computing Systems, CHI'90 Conference Proceedings (Seattle, WA), ACM, New York, April 1990, pp. 337-347.
6. G. Fischer, A. Girgensohn, A.C. Lemke, R. McCall, A. Morch, **Conceptual Frameworks**

and Innovative System Designs for Participatory Design, Proceedings of the Conference on Participatory Design (Seattle, WA), March/April 1990, pp. 59-81.

7. G. Fischer, **Communications Requirements for Cooperative Problem Solving** Systems, The International Journal of Information Systems (Special Issue on Knowledge Engineering), Vol. 15, No. 1, 1990, pp. 21-36.

1989

1. G. Fischer, R. McCall, A. Morch, **JANUS: Integrating Hypertext with a Knowledge-Based Design Environment,** Proceedings of Hypertext'89, ACM, New York, November 1989, pp. 105-117.
2. G. Fischer, **Creativity Enhancing Design Environments,** Proceedings of the International Conference 'Modelling Creativity and Knowledge-Based Creative Design' (Heron Island, Australia), October 1989, pp. 127-132.
3. G. Fischer, R. McCall, A. Morch, **Design Environments for Constructive and Argumentative Design,** Human Factors in Computing Systems, CHI'89 Conference Proceedings (Austin, TX), ACM, New York, May 1989, pp. 269-275.
4. G. Fischer, H. Nieper-Lemke, **HELGON: Extending the Retrieval by Reformulation Paradigm,** Human Factors in Computing Systems, CHI'89 Conference Proceedings (Austin, TX), ACM, New York, May 1989, pp. 357-362.
5. G. Fischer, **Human-Computer Interaction Software: Lessons Learned, Challenges Ahead,** IEEE Software, Vol. 6, No. 1, January 1989, pp. 44-52.

1988

1. G. Fischer, **Cooperative Problem Solving Systems,** Proceedings of the 1st Simposium Internacional de Inteligencia Artificial (Monterrey, Mexico), October 1988, pp. 127-132.
2. G. Fischer, A.C. Lemke, **Construction Kits and Design Environments: Steps Toward Human Problem-Domain Communication,** Human-Computer Interaction, Vol. 3, No. 3, 1988, pp. 179-222.
3. G. Fischer, C. Rathke, **Knowledge-Based Spreadsheet Systems,** Proceedings of AAAI-88, Seventh National Conference of Artificial Intelligence (St. Paul, MN), Morgan Kaufmann Publishers, San Mateo, CA, August 1988, pp. 802-807.

1987

1. G. Fischer, C. Stevens, **Volunteering Information -- Enhancing the Communication Capabilities of Knowledge-Based Systems,** Proceedings of INTERACT'87, 2nd IFIP Conference on Human-Computer Interaction (Stuttgart, FRG), H.-J. Bullinger, B. Shackel (eds.), North-Holland, Amsterdam, September 1987, pp. 965-971.
2. G. Fischer, **Making Computers more Useful and more Usable,** Proceedings of the 2nd International Conference on Human-Computer Interaction (Honolulu, Hawaii), Elsevier Science Publishers, New York, August 1987, pp. 97-104.
3. G. Fischer, **Cognitive View of Reuse and Redesign,** IEEE Software, Special Issue on Reusability, Vol. 4, No. 4, July 1987, pp. 60-72.

I.2 Reports

1992

1. D.F. Redmiles, **From Programming Tasks to Solutions — Bridging the Gap Through the Explanation of Examples,** Ph.D. Dissertation, Department of Computer Science, University of Colorado, 1992.

2. P.W. Foltz, **IRMail: A Minimal Interface for a Retrieval System,** Technical Report, Department of Psychology, University of Colorado, Boulder, CO, 1992.

1991

1. E.C. Ferstl, **Assessment of knowledge structures before and after reading of a text,** Department of Psychology, University of Colorado, Boulder, CO, 1991.

2. P.W. Foltz, **Human Memory Retrieval and Computer Information Retrieval: Similar Approaches to Similar Problems,** Technical Report, Institute of Cognitive Science, University of Colorado, Institute of Cognitive Science, 1991.

3. S.R. Henninger, A.C. Lemke, B.N. Reeves, **A Situated Cognition Perspective on the Design of Cooperative Problem Solving Systems,** Technical Report, Department of Computer Science, University of Colorado, Boulder, CO, April 1991, (Submitted to HCI Journal).

4. S. Henninger, **CODEFINDER: A Tool for Locating Software Objects for Reuse,** Proceedings of AAAI-91 Workshop on Automating Software Design: Interactive Design, AAAI, Anaheim, CA, July 1991, pp. 40-47.

5. S. Henninger, **Human and Computer Task Delegation for Information Retrieval** Systems, Technical Report, Department of Computer Science, University of Colorado, Boulder, CO, 1991.

1990

1. D.F. Redmiles, **Explanation to Support Software Reuse,** Proceedings of the AAAI 90 Workshop on Explanation (Boston, MA), J. Moore, M. Wick (eds.), AAAI, Menlo Park, CA, July 1990, pp. 20-24.

2. S. Henninger, **Defining the Roles of Humans and Computers in Cooperative Problem Solving Systems for Information Retrieval,** Proceedings of the AAAI Spring 1990 Symposium Workshop on Knowledge-Based Human Computer Communication (Palo Alto, CA), G. Fischer, C. Lewis, J. Miller, E. Rich (eds.), AAAI, Menlo Park, CA, March 1990, pp. 46-51.

3. G. Fischer, P.W. Foltz, W. Kintsch, H. Nieper-Lemke, C. Stevens, **Personal Information Systems and Models of Human Memory,** Technical Report, Department of Computer Science, University of Colorado, Boulder, CO, 1990.

1989

1. G. Fischer, W. Kintsch, P.W. Foltz, S.M. Mannes, H. Nieper-Lemke, C. Stevens, **Theories, Methods, and Tools for the Design of User-Centered Systems (Interim Project Report,** September 1986 - February 1989), Technical Report, Department of Computer Science, University of Colorado, Boulder, CO, March 1989.

1987

1. G. Fischer, **Intelligent Support Systems for Hyperknowledge,** Technical Report, Department of Computer Science, University of Colorado, Boulder, CO, November 1987.

I.3 Workshops and HCI Consortium

- **Workshop — Breckenridge 87: Personal Information Systems** — see the following report:

  G. Fischer, H. Nieper (eds.), **Personalized Intelligent Information Systems**, Workshop Report (Breckenridge, CO), Institute of Cognitive Science, University of Colorado, Boulder, CO, Technical Report, No. 87-9, 1987.

  This report includes:

  1. G. Fischer, **Objectives of the Workshop**, Part 1, Chapters 1-4;
  2. W. Kintsch, **Knowledge Assessment and Knowledge Organization**, Chapter 10;
  3. S. Mannes, **Modeling the Generation of Knowledge Structures: The Basics**, Chapter 11;
  4. H. Nieper, **Information Retrieval by Reformulation: From ARGON to HELGON**, Chapter 19.

- **Workshop — Breckenridge 88: Mental Models** — see the following report:

  A.A. Turner (ed.), **Mental Models and User-Centered Design**, Workshop Report (Breckenridge, CO), Institute of Cognitive Science, University of Colorado, Boulder, CO, Technical Report, No. 88-9, 1988.

  This report includes:

  1. G. Fischer, **Mental Models -- A Computer Scientist's Point of View**, pp. 15-26;
  2. P.W. Foltz, W. Kintsch, **An Empirical Study of Retrieval by Reformulation on HELGON**, pp. 9-14.

- **HCI Consortium — Vail 1989: Human-Computer Communication: Innovative Systems and Cognitive Theory**

- **HCI Consortium — San Diego 1990: Information Access**

Appendix II. Graduate Students Supported by the Research Project

Over the duration of this research project, several graduate students have been supported in varying degrees. This appendix provides brief information about their backgrounds and summaries of their Ph.D. research. In each case, the Ph.D. research has been closely linked to the theme of the project. The section titles below include actual or working titles of their dissertations.

II.1 Evelyn Ferstl: Text Comprehension and Readers' Semantic and Sytactic Processes

Advisor: W. Kintsch

Status: Ph.D. Candidate, Department of Psychology

Background: Diplom Mathematics 1987, Ludwig-Maximilians University, Munich, Germany
M.A. Psychology 1991, University of Colorado, Boulder, CO

Abstract:

Text comprehension involves the integration of information from various sources. Lexical, syntactic, and semantic properties of the language, as well as the reader's general world knowledge play an important part in forming representations of a text. My previous research focussed on the interplay of text information and the comprehender's domain knowledge. Using knowledge assessment tasks, both the prior knowledge and the text representation were described in the form of associative networks. The results of two experiments showed that the knowledge assessment tasks were suitable for studying text memory, and that the discourse information was represented in the subjects' knowledge structures. The associative structures obtained after reading could therefore be interpreted as descriptions of the reader's situation model.

Currently under investigation is the issue of how general world knowledge and discourse context influence syntactic processes. In particular, it is still an unresolved question if semantic and pragmatic information is taken into account immediately, or if syntactic processes precede thematic analysis. One approach to distinguishing between these two theoretical accounts is to identify effects of the reader's prior knowledge on syntactic processing.

II.2 Peter Foltz: What can text comprehension theory tell us about Hypertext?

Advisor: W. Kintsch

Status: Ph.D. Candidate, Department of Psychology

Background: B.A. Psychology 1985, Lehigh University, Bethlehem, PA
M.A. Psychology 1988, University of Colorado, Boulder, CO

Abstract:

While there have been claims that hypertext will greatly aid reading comprehension, few studies have shown an advantage in readers' comprehension for hypertext over that of linear text. Thus far, very little theoretical analysis has been done on hypertext. This research used text comprehension theory to com-

pare hypertext to linear text, permitting a comparison of the features of the texts that may aid or hinder the comprehensibility of the text. The background knowledge and goals of the reader were manipulated in order to determine the effect on comprehension and reading strategies. In addition, since hypertext does not always permit coherence when moving from one section to another, a revised hypertext that provided automatic background context to maintain coherence was tested. Results showed that readers of both hypertexts and linear texts use similar reading strategies to navigate through the text. These strategies worked to primarily maintain coherence of the text. Readers of the hypertext used text structure and signals in the text in order to maintain a linear path. These strategies were modeled using the Kintsch (1988) model.

II.3 Scott Henninger: Cognitive Tools for the Location and Comprehension of Software

Advisor: G. Fischer

Status: Ph.D. Candidate, Department of Computer Science

Background: B.S. Electrical Engineering 1983, University of Southern California, Los Angeles, CA
M.S. Computer Science 1990, University of Colorado, Boulder, CO

Abstract:

Cognitive tools for the location and comprehension of software are proposed. Software design is characterized as an ill-defined problem solving process. Example-based programming, a form of software reuse where existing code is modified to meet the current task, is presented as a programming tool. Finding relevant examples in example-based programming systems that have enough examples to be useful presents a problem. Retrieval tools are therefore needed. Traditional information retrieval systems overemphasize retrieval mechanics. Tools are needed to support query construction and relevance evaluation. The theoretical basis for these tools comes from an analysis of human problem solving and memory.

II.4 Suzanne Mannes: Problem-solving as Text Comprehension — A Unitary Approach

Advisor: W. Kintsch

Status: Ph.D. Graduate, 1989, Department of Psychology

Background: B.A. 1982, State University of New York, Plattsburgh, NY
M.A. Psychology 1986, University of Colorado, Boulder, CO

Abstract:

A system called NETWORK is described which implements the construction-integration model of Kintsch (1988) in a routine computing task domain. This system builds a plan of action on-line for a given task from a set of plan-elements. These plan-elements are simple over-learned production rules which are put together by NETWORK to produce plans for novel tasks.

NETWORK takes as input a task description, uses this information to select related knowledge from its long-term member, and *constructs* a network representation of the task. This network is then *integrated* through a spreading activation procedure where irrelevant items in the network become deactivated, and things which appear related, sustain each other's higher activation. Subsequently, a decision process chooses a plan-element for firing, depending upon its level of activation. Plan-elements which are more highly activated are considered for action first. When a plan-element is found which can fire, its outcomes are added to the state of the world. The process repeats until a selection of plan-elements is produced which completes the task.

NETWORK solves several computer tasks on which it was developed, synthesizing plans like planning systems. Although the construction-integration model was intended as a theory of text comprehension, it displays planning behavior in the instances presented here. This cross domain application of a single model may lead us closer to identifying unifying themes in cognition.

II.5 David Redmiles: From Programming Tasks to Solutions — Bridging the Gap through the Explanation of Examples

Advisor: G. Fischer

Status: Ph.D. Graduate, 1992, Department of Computer Science

Background: B.S. Mathematics and Computer Science 1980,
M.S. Computer Science 1982,
The American University, Washington, D.C.

Abstract:

Evidence, experience, and observation indicate that examples provide a powerful aid for problem solvers. In the domain of software engineering, examples not only provide objects to be reused but also a context in which users can explore issues related to the current task. This dissertation describes a software tool called EXPLAINER, which supports programmers' use of examples in the domain of graphics programming, assisting them with examples and explanations from various views and representation perspectives. EXPLAINER provides a conceptual and working framework for the study of programmers' uses of examples in problem solving and serves as a test bed for representations based upon multiple perspectives. The EXPLAINER approach is evaluated and compared with other available approaches, such as on-line manuals. The evaluation showed that subjects using EXPLAINER exhibited a more controlled and directed problem-solving process compared to subjects using a commercially available, searchable on-line manual. Representation of examples from multiple perspectives is seen as a critical aspect of catalog-based design environments.

II.6 Curt Stevens: Information Access in Complex, Poorly Structured Information Spaces

Advisor: G. Fischer

Status: Ph.D. Candidate, Department of Computer Science

Background: B.A. Economics 1984, University of California, Berkeley, CA
M.S. Computer Science 1989, University of Colorado, Boulder, CO

Abstract:

Large information spaces present several problems including information overload. This research effort focuses on the domain of Usenet News, an open access, computer-based bulletin board system, which distributes messages and software. A conceptual framework is developed that shows the need for (a) flexible organization of information access interfaces, (b) personalized structure to deal with vocabulary mismatches, and (c) semi autonomous agents that assist in creating this personalized structure. An operational innovative system building effort (INFOSCOPE) instantiates the framework. In INFOSCOPE, users can evolve the predefined system structure to suit their own semantic interpretations. The approach taken by INFOSCOPE differs from other approaches by requiring less up-front structuring by message senders and allowing users to be filter critics instead of filter creators.

Appendix III. Additional Information about the Research Project

III.1 Professional Researchers working with the Project

- Helga Nieper-Lemke, Research Associate, Department of Computer Science, 1986-1989
- Thea Turner, Post-Doctoral Fellow, Institute of Cognitive Science, 1986-1987, (current position: Member Technical Staff, Nynex Science and Technology Center)

III.2 External Collaborations

William Moninger, NOAA, Boulder, CO. William Moninger has been working in the METALOG system, a personal intelligent information system for the management of scientific "metadata," that is, data about scientific data. The initial system is being developed for use by the radar and lidar program areas of the Wave Propagation Laboratory, to be used in conjunction with their new data analysis workstation. The system should be applicable, however, to any scientific research that involves detailed study of large amounts of data displayed on a computer.

Hans Brunner, USWEST, Denver, CO. Hans Brunner, Scott Wolff, and Andy Parng have been working on various systems in the Intelligent Customer Assistance project. In particular, Andy has implemented the IDEAS system, a query system based upon the retrieval by reformulation paradigm. IDEAS extends the work we have done on the HELGON system by adding a new query specification tool. This tool allows users to specify parts of the query in a graphical manner. For example, to find all houses for sale in a two mile radius of a certain point the user specifies that the subject of the initial query is homes for sale. He then draws a two mile circle on a map provided by the system. The display then indicates where any candidate homes are by flashing them on the map. At this point the user can zoom in on the map to make a more specific graphical query, or can reformulate the other part of the query. This might include a specification of the acceptable price range for the home search in question. In this way, there is very little effort necessary on the part of the user in order to evolve the specification from their situation model to the system model (assuming that the user knows how to read a map).

Mike Atwood, NYNEX, White Plains, NY. NYNEX (as other companies) faces a number of problems where the research efforts within our project offers interesting ways to tackle some of their major problems. For example: their large information spaces consist for example of millions of COBOL programs which were written over the last 20 years. There is nobody around who understands these information spaces any more. These spaces are heterogeneous and lack a good conceptual structure. Traditional database approaches have badly failed in tackling the problem of maintaining and updating these information stores.

Thomas Landauer, Bellcore, Morristown, NJ. The research group at Bellcore under the direction of Tom Landauer has investigated a number of interesting problems which are directly relevant to our research project (e.g. the vocabulary problem [Furnas 86] and semantic retrieval [Deerwester 90] Peter Foltz, who works on the ARI project, did a year long internship at Bellcore. This has provided us with a unique opportunity to intensify our research collaboration with this group. We have used their Latent Semantic Indexing retrieval methods in order to investigate some aspects of retrieval.

Ron Brachman, AT&T Bell Laboratories, Murray Hill, NJ. Several researchers (e.g., Ron Brachman and Peter Patel-Schneider) at AT&T Bell Laboratories were major contributors to the ARGON system

which we got from them several years ago and which served as the starting point for the HELGON system. The problems which they encounter are very similar to the problems described above for NYNEX. They are working on several new systems (e.g., a successor to the KANDOR knowledge representation formalism and a new version of an ARGON like system) which are of direct relevance to the efforts in our project. They obtained from us (in return for giving us the original version of the ARGON system): the HELGON system, the HELGON tape and a new version of ARGON (converted by us from Release 6 to Release 7 on the SYMBOLICS).

Erich Neuhold, GMD-F4, Darmstadt, W-Germany. The research group in Darmstadt works in the general area of "integrated publication and information systems". In the context of a cooperation agreement between the Computer Science Department and the Institute of Cognitive Science at the University of Colorado, Boulder and them, the topics of our ARI project have played an important role. We provided them with a copy of the HELGON system and the HELGON tape.

Appendix IV. Assessment of Relevance to ARI and the Army

Thomas W. Mastaglio and James Sullivan assisted us by assessing the relevance of this research project to the ARI and the U.S. Army. Their comments are included below. Both are researchers who are simultaneously familiar with the needs of the Army and our work.

IV.1 Thomas W. Mastaglio: An Assessment of the Applicability of the Research Work

Remark: The following note was written in January 1990 by Thomas W. Mastaglio who, at that time, was a Graduate Student at the University of Colorado and a Lieutenant Colonel in the U.S. Army.

The research project "Design of User-Centered Computing Systems" has the potential to provide guidelines to Department of Defense agencies involved in systems development. Tactical and administrative systems could both benefit from the theory and specific technologies that are coming out of this work. At the conceptual level, many military applications are involved in access to large information spaces. Their full use is limited not by computational power but by the ability of users to find and use what they need.

Often the conditions under which systems must be used include severe time constraints, harsh environments and high stress, operational combat situations. Under these circumstances, the capabilities of working memory and other recognized limitations of human cognition are even more severely restricted. The introduction of increasing numbers of computer systems to "aid" decision makers and their staffs creates a plethora of what these researchers call "high functionality systems" in operational environments. Such Army systems as TACFIRE (the Field Artillery targeting and engagement management system) and the Maneuver Control System (MCS) are already present examples. Their successors will introduce even more complexity at the user interface level and further tax available human cognitive capacities.

Staff officers in fields such as operations planning, intelligence analysis, and budgeting are often forced to use large databases from diverse sources. To access these information stores users are currently most often required to use what this research calls the "reformulation approach". New systems developed to aid users should be designed from a "situation model" perspective. I would conjecture that the result would be significant improvements in speed, quality, and breadth of their work.

I was briefed on one system during a recent visit to the Army's Training and Doctrine Command (TRADOC), the *Asset Inventory Analyzer*, that attempts to incorporate many of the ideas investigated in this project. It is not coincidence that one of the primary developers in the TRADOC Artificial Intelligence Cell, Captain Jim Sullivan, recently completed an MS in Computer Science at the University of Colorado. *Asset Inventory Analyzer* is a "system for experts" not an expert system. It is used by force developers to reason about the procurement of high cost weapons systems and their projected effect on individual unit readiness ratings for the next 20 years. The initial prototype was developed for aircraft modernization planning. The system uses a direct manipulation interface that allows the user to conduct open ended "what if" analysis, a task previously done using pencil, paper and off-line database printouts. An associated business graphics package encapsulates the results of changes to data and the resulting analysis in graphical form. This system is a tool that, in addition to providing analysis functions, shows the user information using multiple techniques; it serves as an extension of the users' working memory.

The aviation version is in use at Department of Army staff level. I do not recall specific data but the developers claim a significant improvement in both the quality of the users work and their ability to brief it to decision makers.

During some other visits I recently made and meetings I attended other agencies also expressed an interest in work on AI in Human-Computer Interaction at the University of Colorado:

- The AI Center for the Army Staff in the Pentagon showed me a system called SABRE that is similar to the Assets Inventory Analyzer. It considers all reportable lines of equipment and helps project readiness conditions for entire commands. They have had great success in using AI and some current user interface technologies to provide the staff planner access to a tremendous amount of data. The major shortcoming (by my assessment) seems to be in filtering and summarizing that information in order to reduce the complexity confronting the user. I had some interesting discussion with one of the developers of SABRE about how they could assess how well the system actually fulfills user needs and then apply current ideas from this research to answer those needs.

- The US Army Missile Command is developing a coach like trainer for managing air assets on the battlefield. They feel that they need an individual user modelling capability in order for their system to be effective. They are quite interested in the approaches to user modelling being worked on at the University of Colorado.

- The Army's Project Manager for Training Devices (PM-TRADE) do not have good models for what should go into a system design when considering the use of modern technologies. An "Intelligent Tutor" for rifle marksmanship was contracted for and delivered before anyone realized that this was probably not an appropriate application of that technology. They need advice and models for analyzing user needs *before* selecting an approach — models for how to design user-centered systems.

The analysis and comments contained in this short assessment are strictly a personal view. They do reflect an official Department of Defense position and should *not* be represented as such.

IV.2 James Sullivan: An Assessment of the Applicability of the Research Work

Remark: The following note was written in January 1992 by James Sullivan who at that time was a Graduate Student at the University of Colorado and a Major in the U.S. Army.

There are several salient aspects from the theory, methods, and tools for the design of user-centered computer systems that are of potential interest and application to Department of Defense agencies. In this brief assessment, I will focus on current defense needs that I feel are worth noting and then remark on how the research work presented here is applicable to these needs. While these observations are primarily based on my experiences with building tools for aviation and armor force modernization planning, I believe that they are also potentially applicable to other DOD areas, such as contingency planning, logistics management, force structuring, resource allocation, and program management.

Defense Needs. If there is any lesson to be learned from recent history, it is that planning for change has become be the norm, and not the exception. The ability of a DOD staff or agency to dynamically react to rapidly changing international and domestic events will be the true measure of success or failure. As our defense forces become leaner and move from bases in overseas theaters to CONUS, compressed time and logistical constraints will prevent us from designing a solution to a problem "from scratch" every time something unexpected arises. We will have to become even more adept at piecing together and modifying satisfactory plan components while readily identifying and retooling that which is inappropriate

or outdated. We will also need to make use of lessons learned from our past experiences so that new plans are robust and comprehensive.

Planning will also continue to be a parallel and multi-dimensional activity. Over the past 24 months, for example, our defense services have had to simultaneously plan for the transition from peace to war, the execution of a war campaign in a foreign theater, the restructuring of our combat forces, the restationing of troops, and finally, the downsizing and demobilization of our armed forces. Although the execution of some of these activities were contingent upon the completion of others, the planning for all of them was in parallel and used similar information about force structure, equipment on hand, personnel, and future modernization schedules.

Finally, because of the tremendous downsizing we are facing in the post Cold War era, many civilian and military experts will probably not be available to explain all rationale behind past plans and decisions. Our institutions need to capture not only "how to" knowledge and expertise, but also the "why" knowledge or it will be forever lost.

Relevance of the Theory. Unlike the traditional expert systems approach to automation, this theory asserts that people should not be simply replaced with autonomous decision-making systems, but empowered with a synergistic computational environment for designing solutions to complex and dynamic problems. This theory is very applicable in light of the current and projected budget cutting trends that will leave staffs with fewer personnel to manage new and ever changing complexities of defense planning. While expert systems technology is certainly mature and well understood, domains such as contingency planning, logistics management, force structuring, resource allocation, and program management are all too complex and dynamic to be adequately captured by a suite of expert systems. However, one could conceive how the development of a set of intelligent modular planning tools could be very useful to several defense agency analysts in a variety of these problem domains.

Since this theory articulates the power of a human-machine team toward a solution, it is important to recognize and develop appropriate cognitive theory to address the strengths and limitations of the human user in such an environment. This theory is extremely relevant in view of the proliferation of DOD data bases and information sources. The challenge in the future will not be to create new stores of information, but to use that which has already been created, and this theoretical research is both insightful and useful in approaching this problem.

A final aspect of this theory that is relevant to our need to capture and use expertise is reflected in how rationale is both stored and used in the design process. Artifacts that are created with these tools are not only recognized for the solution they present to the problem at hand, but also for the insight and argumentation they provide to future solutions. This is significant because it demonstrates a very viable way to acquire and accumulate functional expertise in a way that will provide a meaningful context for future reuse, explanation and learning.